

# **MULTI-MODAL BROWSING AND METHOD AND SYSTEM FOR EXECUTING CONVERSATION TYPE MARK-UP LANGUAGE**

**Publication number:** JP2001154852 (A)

**Publication date:** 2001-06-08

**Inventor(s):** MAES STEPHANE HERMAN; RAMAN THIRUVILVAMA LAI V

**Applicant(s):** IBM

**Classification:**


- **international:** **G06F9/45; G06F15/00; G06F9/45; G06F15/00;** (IPC1-7): G06F9/45

- **European:**

**Application number:** JP20000311661 20001012

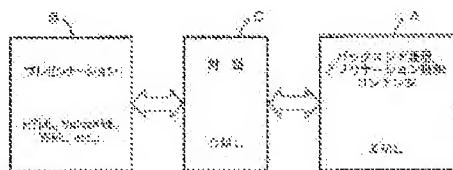
**Priority number(s):** US19990158777P 19991012; US20000544823 20000406

**Also published as:**

 JP4140878 (B2)

Abstract of **JP 2001154852 (A)**

**PROBLEM TO BE SOLVED:** To provide new application programming language based on user interaction with an arbitrary device which is being used by a user for performing access to information in an arbitrary type. **SOLUTION:** In a desired execution configuration, this conversation type mark-up language(CML) is the language of a high level XML base for expressing a 'dialog' or 'conversation' to be performed by a user with a prescribed computing device. An application preparer can program an application by using the element of an interactive base called 'conversation type gesture'. Also, it is possible to realize the various execution configurations of a multi-modal browser for supporting the characteristics of the CML according to various modal specific expressions, for example, the graphical user interface(GUI) browser of an HTML base and the speech browser of a Voice XML base.



-----  
Data supplied from the **esp@cenet** database — Worldwide

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号  
特開2001-154852  
(P2001-154852A)

(43) 公開日 平成13年6月8日 (2001.6.8)

(51) Int.Cl. <sup>7</sup> G 0 6 F 9/45	識別記号	F I C 0 6 F 9/44	テーマコード* (参考) 3 2 0 A
---	------	---------------------	-------------------------

審査請求 有 請求項の数91 O L (全 85 頁)

(21) 出願番号 特願2000-311661 (P2000-311661)

(22) 出願日 平成12年10月12日 (2000. 10. 12)

(31) 優先権主張番号 6 0 / 1 5 8 7 7 7

(32) 優先日 平成11年10月12日 (1999. 10. 12)

(33) 優先権主張国 米国 (US)

(31) 優先権主張番号 0 9 / 5 4 4 8 2 3

(32) 優先日 平成12年4月6日 (2000. 4. 6)

(33) 優先権主張国 米国 (US)

(71) 出願人 390009531

インターナショナル・ビジネス・マシーンズ・コーポレーション  
INTERNATIONAL BUSINESS MACHINES CORPORATION  
アメリカ合衆国10504、ニューヨーク州  
アーモンク (番地なし)

(74) 代理人 100086243

弁理士 坂口 博 (外2名)

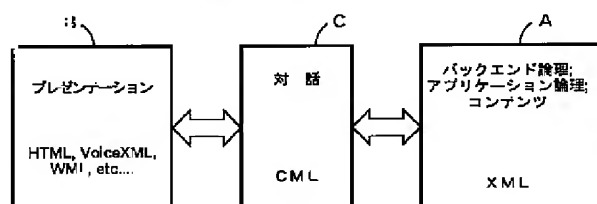
最終頁に続く

(54) 【発明の名称】 マルチモーダル・ブラウジングおよび会話型マークアップ言語の実施の方法およびシステム

(57) 【要約】 (修正有)

【課題】 ユーザが任意のタイプの情報にアクセスするために利用している任意の装置とのユーザ対話に基づく、新しいアプリケーション・プログラミング言語を提供すること。

【解決手段】 この新しい言語は、本明細書で「会話型マークアップ言語 (CML)」と呼ぶ。好ましい実施形態では、CMLは、ユーザが所与のコンピューティング装置と行うことになる「ダイアログ」または「会話」を表すためのハイレベルXMLベースの言語である。アプリケーション作成者は、「会話型ジェスチャ」と呼ぶ対話ベースの要素を使用してアプリケーションをプログラムすることができる。本発明はまた、様々なモーダル性特定の表現、例えばHTMLベースのグラフィカル・ユーザ・インタフェース (GUI) ブラウザ、VoiceXMLベースのスピーチ・ブラウザなどに従ってCMLの特徴をサポートすることのできるマルチモーダル・ブラウザの様々な実施形態を可能にする。



# 【特許請求の範囲】

【請求項1】 ユーザから1つまたは複数のコンピュータ・ベースの装置を介してアクセス可能なアプリケーションをプログラムする方法であって、

前記アプリケーションにアクセスするために使用される前記1つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話を表現するステップを含み、

前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立、さらに、コンポーネントごとにトランスコーディングされて、前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの1つまたは複数のモジュール性特定のレンダリングを生成する方法。

【請求項2】 前記アプリケーションの少なくとも一部がサーバから、クライアントの役割を果たす前記1つまたは複数のコンピュータ・ベースの装置のうちの少なくとも1つにダウンロードされるクライアント／サーバ構成において、前記サーバに常駐する前記コンテンツ／アプリケーション論理への接続を提供するように動作可能なコードを前記アプリケーション中に含めるステップをさらに含む、請求項1に記載の方法。

【請求項3】 前記コンテンツ／アプリケーション論理接続コードが、前記アプリケーションに関連する1つまたは複数のデータ・モデル、属性制約、および妥当性検査規則のうちの少なくとも1つを表す、請求項2に記載の方法。

【請求項4】 前記1つまたは複数のモジュール性特定のレンダリングが、前記アプリケーションの一部のスピーチ・ベースの表現を含む、請求項1に記載の方法。

【請求項5】 前記スピーチ・ベースの表現がVoiceXMLに基づく、請求項4に記載の方法。

【請求項6】 前記1つまたは複数のモジュール性特定のレンダリングが、前記アプリケーションの一部の視覚ベースの表現を含む、請求項1に記載の方法。

【請求項7】 前記視覚ベースの表現がHTML、CHTML、WMLのうちの少なくとも1つに基づく、請求項6に記載の方法。

【請求項8】 前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的に表現される、請求項1に記載の方法。

【請求項9】 前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって命令的に表現される、請求項1に記載の方法。

【請求項10】 前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的かつ命令的に表現される、請求項1に記載の方法。

【請求項11】 前記対話ベースのプログラミング・コン

ポーネントが、前記ユーザと前記1つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する基本要素を含む、請求項1に記載の方法。

【請求項12】 前記対話ベースのプログラミング・コンポーネントが複合要素を含み、前記複合要素は、前記ユーザと前記1つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する2つ以上の前記基本要素の集合体である、請求項11に記載の方法。

【請求項13】 前記対話ベースのプログラミング・コンポーネントの1つが会話型ジェスチャを表す、請求項1に記載の方法。

【請求項14】 前記会話型ジェスチャが、ユーザへの情報メッセージをカプセル化するジェスチャを含む、請求項13に記載の方法。

【請求項15】 前記会話型ジェスチャが、コンテキスト・ヘルプ情報をカプセル化するジェスチャを含む、請求項13に記載の方法。

【請求項16】 前記会話型ジェスチャが、別のジェスチャの完了が成功したときに行われるアクションをカプセル化するジェスチャを含む、請求項13に記載の方法。

【請求項17】 前記会話型ジェスチャが、イエス／ノー・ベースの質問をカプセル化するジェスチャを含む、請求項13に記載の方法。

【請求項18】 前記会話型ジェスチャが、ユーザが選択肢のセットから選択することを期待される場合のダイアログをカプセル化するジェスチャを含む、請求項13に記載の方法。

【請求項19】 前記選択ジェスチャが前記選択肢のセットを表すサブ要素を含む、請求項18に記載の方法。

【請求項20】 前記選択ジェスチャが、選択がパスすべきテストを表すサブ要素を含む、請求項18に記載の方法。

【請求項21】 前記選択ジェスチャが、前記テストが不合格の場合に呈示すべきエラー・メッセージを表すサブ要素を含む、請求項20に記載の方法。

【請求項22】 前記会話型ジェスチャが、所与の会話型ジェスチャの結果を妥当性検査するための規則をカプセル化するジェスチャを含む、請求項13に記載の方法。

【請求項23】 前記会話型ジェスチャが、文法処理規則をカプセル化するジェスチャを含む、請求項13に記載の方法。

【請求項24】 前記会話型ジェスチャが、ユーザが前記アプリケーションの各部分をナビゲートするのを助けるダイアログをカプセル化するジェスチャを含む、請求項13に記載の方法。

【請求項25】 前記会話型ジェスチャが、少なくとも1つのユーザ・ログインおよび認証の情報を求める要求をカプセル化するジェスチャを含む、請求項13に記載の方法。

方法。

【請求項26】前記会話型ジェスチャが、制約付きのユーザ入力を求める要求をカプセル化するジェスチャを含む、請求項13に記載の方法。

【請求項27】前記会話型ジェスチャが、制約のないユーザ入力を求める要求をカプセル化するジェスチャを含む、請求項13に記載の方法。

【請求項28】前記会話型ジェスチャが、情報のサブミットを制御するジェスチャを含む、請求項13に記載の方法。

【請求項29】論理入力イベント、ならびに、前記論理入力イベントと定義された前記論理入力イベントをトリガする物理入力イベントとの間の関連を定義する機構を提供するステップをさらに含む、請求項1に記載の方法。

【請求項30】前記コンポーネントごとのトランスコーディングがXSL変換規則に従って行われる、請求項1に記載の方法。

【請求項31】前記コンポーネントごとのトランスコーディングがJava Beanに従って行われる、請求項1に記載の方法。

【請求項32】前記コンポーネントごとのトランスコーディングがJava Server Pageに従って行われる、請求項1に記載の方法。

【請求項33】前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが、前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記1つまたは複数のモーダル性特定のレンダリングを同期させることを可能にする、請求項1に記載の方法。

【請求項34】前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが自然言語理解環境をサポートする、請求項1に記載の方法。

【請求項35】前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記1つまたは複数のモーダル性特定のレンダリングに関連するプレゼンテーション・フィーチャを表面的に変更することを可能にするコードを含めるステップをさらに含む、請求項1に記載の方法。

【請求項36】コンポーネントごとにトランスコーディングして前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記1つまたは複数のモーダル性特定のレンダリングを生成するための規則を変更することを可能にするコードを含めるステップをさらに含む、請求項1に記載の方法。

【請求項37】実装されている、基礎をなすデータ・モデルの定義が、前記ユーザ対話を定義するマークアップ言語から分離される、請求項1に記載の方法。

【請求項38】node\_id属性が各コンポーネントに付加され、前記属性は様々な出力の上にマッピングされる、

請求項1に記載の方法。

【請求項39】モーダル性特定のマークアップ・コンポーネントをカプセル化する通過機構が作成者に提供される、請求項1に記載の方法。

【請求項40】前記コンポーネントが並列に活動化させることができる、請求項1に記載の方法。

【請求項41】前記プレゼンテーションおよびトランスコーディングが拡張可能である、請求項1に記載の方法。

【請求項42】前記アプリケーションの状態がカプセル化される、請求項1に記載の方法。

【請求項43】前記表現が動的に生成されるデータの参照を可能にし、前記コンテンツ／アプリケーション論理へのコールバック機構をサポートする、請求項1に記載の方法。

【請求項44】1つまたは複数のコンピュータ・ベースの装置に関連するアプリケーションにアクセスする際に使用する装置であって、

1つまたは複数のプロセッサを備え、前記プロセッサが、(i)アプリケーション・サーバから前記アプリケーションを得るように動作可能であり、前記アプリケーションが、前記1つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話によってプログラマ的に表現され、前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立し、前記プロセッサはまた、(ii)前記対話ベースのプログラミング・コンポーネントをコンポーネントごとにトランスコーディングして、前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの1つまたは複数のモーダル性特定のレンダリングを生成するように動作可能である装置。

【請求項45】前記1つまたは複数のプロセッサが前記1つまたは複数のコンピュータ・ベースの装置にわたって分散される、請求項44に記載の装置。

【請求項46】前記アプリケーションの少なくとも一部がサーバから、クライアントの役割を果たす前記1つまたは複数のコンピュータ・ベースの装置のうちの少なくとも1つにダウンロードされるクライアント／サーバ構成において、前記サーバに常駐する前記コンテンツ／アプリケーション論理への接続を提供するように動作可能なコードを前記アプリケーション中に含めるステップをさらに含む、請求項44に記載の装置。

【請求項47】前記コンテンツ／アプリケーション論理接続コードが、前記アプリケーションに関連する1つまたは複数のデータ・モデル、属性制約、および妥当性検査規則のうちの少なくとも1つを表す、請求項46に記載の装置。

【請求項48】前記1つまたは複数のモーダル性特定の



レンダリングが、前記アプリケーションの一部のスピーチ・ベースの表現を含む、請求項44に記載の装置。

【請求項49】前記スピーチ・ベースの表現がVoiceXMLに基づく、請求項48に記載の装置。

【請求項50】前記1つまたは複数のモーダル性特定のレンダリングが、前記アプリケーションの一部の視覚ベースの表現を含む、請求項44に記載の装置。

【請求項51】前記視覚ベースの表現がHTML、CHTML、WMLのうちの少なくとも1つに基づく、請求項50に記載の装置。

【請求項52】前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的に表現される、請求項44に記載の装置。

【請求項53】前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって命令的に表現される、請求項44に記載の装置。

【請求項54】前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的かつ命令的に表現される、請求項44に記載の装置。

【請求項55】前記対話ベースのプログラミング・コンポーネントが、前記ユーザと前記1つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する基本要素を含む、請求項44に記載の装置。

【請求項56】前記対話ベースのプログラミング・コンポーネントが複合要素を含み、前記複合要素は、前記ユーザと前記1つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する2つ以上の前記基本要素の集合体である、請求項55に記載の装置。

【請求項57】前記対話ベースのプログラミング・コンポーネントの1つが会話型ジェスチャを表す、請求項44に記載の装置。

【請求項58】前記会話型ジェスチャが、ユーザへの情報メッセージをカプセル化するジェスチャを含む、請求項57に記載の装置。

【請求項59】前記会話型ジェスチャが、コンテキスト・ヘルプ情報をカプセル化するジェスチャを含む、請求項57に記載の装置。

【請求項60】前記会話型ジェスチャが、別のジェスチャの完了が成功したときに行われるアクションをカプセル化するジェスチャを含む、請求項57に記載の装置。

【請求項61】前記会話型ジェスチャが、イエス／ノー・ベースの質問をカプセル化するジェスチャを含む、請求項57に記載の装置。

【請求項62】前記会話型ジェスチャが、ユーザが選択肢のセットから選択することを期待される場合のダイアログをカプセル化するジェスチャを含む、請求項57に記載の装置。

【請求項63】前記選択ジェスチャが前記選択肢のセッ

トを表すサブ要素を含む、請求項62に記載の装置。

【請求項64】前記選択ジェスチャが、選択がパスすべきテストを表すサブ要素を含む、請求項62に記載の装置。

【請求項65】前記選択ジェスチャが、前記テストが不合格の場合に呈示すべきエラー・メッセージを表すサブ要素を含む、請求項64に記載の装置。

【請求項66】前記会話型ジェスチャが、所与の会話型ジェスチャの結果を妥当性検査するための規則をカプセル化するジェスチャを含む、請求項57に記載の装置。

【請求項67】前記会話型ジェスチャが、文法処理規則をカプセル化するジェスチャを含む、請求項57に記載の装置。

【請求項68】前記会話型ジェスチャが、ユーザが前記アプリケーションの各部分をナビゲートするのを助けるダイアログをカプセル化するジェスチャを含む、請求項57に記載の装置。

【請求項69】前記会話型ジェスチャが、少なくとも1つのユーザ・ログインおよび認証の情報を求める要求をカプセル化するジェスチャを含む、請求項57に記載の装置。

【請求項70】前記会話型ジェスチャが、制約付きのユーザ入力を求める要求をカプセル化するジェスチャを含む、請求項57に記載の装置。

【請求項71】前記会話型ジェスチャが、制約のないユーザ入力を求める要求をカプセル化するジェスチャを含む、請求項57に記載の装置。

【請求項72】前記会話型ジェスチャが、情報のサブミットを制御するジェスチャを含む、請求項57に記載の装置。

【請求項73】論理入力イベント、ならびに、前記論理入力イベントと定義された前記論理入力イベントをトリガする物理入力イベントとの間の関連を定義するための機構を提供するステップをさらに含む、請求項44に記載の装置。

【請求項74】前記コンポーネントごとのトランスコーディングがXSL変換規則に従って行われる、請求項44に記載の装置。

【請求項75】前記コンポーネントごとのトランスコーディングがJava Beanに従って行われる、請求項44に記載の装置。

【請求項76】前記コンポーネントごとのトランスコーディングがJava Server Pageに従って行われる、請求項44に記載の装置。

【請求項77】前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが、前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記1つまたは複数のモーダル性特定のレンダリングを同期させることを可能にする、請求項44に記載の装置。

【請求項78】前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが自然言語理解環境をサポートする、請求項44に記載の装置。

【請求項79】前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記1つまたは複数のモーダル性特定のレンダリングに関連するプレゼンテーション・フィチャを表面的に変更することを可能にするコードを含めるステップをさらに含む、請求項44に記載の装置。

【請求項80】コンポーネントごとにトランスコーディングして前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記1つまたは複数のモーダル性特定のレンダリングを生成するための規則を変更することを可能にするコードを含めるステップをさらに含む、請求項44に記載の装置。

【請求項81】実装されている、基礎をなすデータ・モデルの定義が、前記ユーザ対話を定義するマークアップ言語から分離される、請求項44に記載の装置。

【請求項82】node\_id属性が各コンポーネントに付加され、前記属性は様々な出力の上にマッピングされる、請求項44に記載の装置。

【請求項83】モーダル性特定のマークアップ・コンポーネントをカプセル化する通過機構が作成者に提供される、請求項44に記載の装置。

【請求項84】前記コンポーネントが並列に活動化されることができ、請求項44に記載の装置。

【請求項85】前記プレゼンテーションおよびトランスコーディングが拡張可能である、請求項44に記載の装置。

【請求項86】前記アプリケーションの状態がカプセル化される、請求項44に記載の装置。

【請求項87】前記表現が動的に生成されるデータの参照を可能にし、前記コンテンツ／アプリケーション論理へのコールバック機構をサポートする、請求項44に記載の装置。

【請求項88】前記1つまたは複数のプロセッサが前記1つまたは複数のコンピュータ・ベースの装置にわたって分散され、前記アプリケーションが前記1つまたは複数のコンピュータ・ベースの装置にまたがって同期化される、請求項44に記載の装置。

【請求項89】前記アプリケーションの表現がさらに、1つまたは複数のモーダル性特定のマークアップ言語を介して前記1つまたは複数のモーダル性特定のレンダリングを表面的に変更することを可能にする、請求項44に記載の装置。

【請求項90】ユーザからの1つまたは複数のコンピュータ・ベースの装置を介したアプリケーションへのアクセスを提供する際に使用するブラウザ装置であって、コンピュータ実行可能なコードを含む機械読取可能媒体を備え、前記コンピュータ実行可能なコードが、実行時

に、

アプリケーション・サーバからアプリケーションを得るステップであって、前記アプリケーションが、前記1つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話によってプログラマ的に表現され、前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立しているステップと、

前記対話ベースのプログラミング・コンポーネントをコンポーネントごとにトランスコーディングして、前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの1つまたは複数のモーダル性特定のレンダリングを生成するステップとを実施することを可能にするブラウザ装置。

【請求項91】ユーザから1つまたは複数のコンピュータ・ベースの装置を介してアクセス可能なアプリケーションをプログラミングする際に使用する製造品であって、コンピュータ実行可能なコードを含む機械読取可能媒体を備え、前記コンピュータ実行可能なコードが、実行時に、

前記アプリケーションにアクセスするために使用される前記1つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話を表現するステップを実施することを可能にし、

前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立、さらに、コンポーネントごとにトランスコーディングされて、前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの1つまたは複数のモーダル性特定のレンダリングを生成する製造品。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は一般に情報アクセス・アプリケーションに関し、より詳細には、そのような情報アクセス・アプリケーションで使用するための、対話ベースのマークアップ言語および対話ベースのマークアップ言語をサポートするマルチモーダル・ブラウジング機構に関する。

【0002】

【従来の技術】関連出願の相互参照

本発明は、1999年10月12日出願の米国仮特許出願第60/158777号への優先権を主張し、この開示を参照により本明細書に組み込む。本明細書は、

(i) US99/23008 (整理番号YO998-392) として識別される1999年10月1日出願のPCT国際特許出願、(ii) US99/22927 (整理

番号Y0999-111)として識別される1999年10月1日出願のPCT国際特許出願、(iii)US99/22925(整理番号Y0999-113)として識別される1999年10月1日出願のPCT国際特許出願に関係し、上記の各PCT国際特許出願は、1998年10月2日出願の米国仮特許出願第60/102957号および1999年1月27日出願の米国仮特許出願第60/117595号への優先権を主張する。さらに本明細書は、(iv)2000年2月18日出願の米国特許出願第09/507526号(整理番号Y0999-178)に関係し、この出願は、1999年4月7日出願の米国仮特許出願第60/128081号および1999年10月12日出願の米国仮特許出願第60/158777号への優先権を主張する。上に参照した関連出願のすべてを参照により本明細書に組み込む。

【0003】様々なタイプおよび量の情報の使用可能性が劇的に上昇し、そのような情報にアクセスする従来の機構の時間または使用可能性あるいはその両方が急激に減少したために、個人は現在、いっような装置からでも、どんな情報に対してもアクセスまたは作用または変換できるようになりたい、あるいはそのすべてを行えるようになりたいと思っている。例えばインターネットの場合、様々な大量の情報が利用可能だが、インターネットは従来、HTTP(HyperText Transport Protocol)ネットワークの最上部にあるHTML(HyperText Markup Language)ブラウザを使用して情報にアクセスする装置しか主にサポートしていなかった。これは、TCP/IP(Transmission Control Protocol/Internet Protocol)の最上部に提供されていた。

【0004】この問題に対する解決法は、そのような情報にアクセスするのに使用されるアプリケーション・プログラムを書き直し、それによって他の方法でその情報にアクセスできるようにすることが中心であった。ある解決法は、WAP(Wireless Application Protocol)の開発へとつながった。<http://www.mobilewap.com>を参照されたい。WAPは、無線ネットワーク用のHTTPに相当する。無線ネットワーク用のHTMLに相当するWML(Wireless Markup Language)が開発された。したがって、HTTPの最上部でHTMLが使用される方式と同様に、WAPの最上部でWMLが使用される。WAPおよびWMLにより、ユーザは、制約された画面レンダリングおよび限られた帯域幅の接続能力を有するセルラー・ホンを通じてインターネットにアクセスすることができる。CHTMLが、この空間を対象とするML(マークアップ言語)のもう1つの例である。

【0005】次に、より最近になり、ウェブ・プログラミング・モデル(ファット・クライアント・プログラミング・モデルとも呼ばれる)をボイス・アクセス、特に電話アクセスおよび自動音声応答(IVR)システムに導入するための機構が開発された。このような機構は、

通常、スピーチ・ブラウザ(またはボイス・ブラウザ)と呼ばれる。このようなスピーチ・ブラウザは、上に参照した、米国出願第60/102957号(整理番号Y0998-382)として識別される米国仮特許出願に記載されている。スピーチ・ブラウザは、VoiceXMLと呼ばれる、XML(Extensible Markup Language)のスピーチ・ベースの変形を使用することができる。<http://www.voicexml.org>を参照されたい。スピーチ・ブラウザはまた、WAPプロトコルの最上部でWMLデータの交換と共に動作することもできる。

【0006】しかし、このような手法は、アプリケーション・プログラマがマルチチャネル・サポートを提供したい場合、すなわち、ウェブ・ブラウザ(HTMLブラウザ)、電話(ボイス・ブラウザ)、および無線ブラウザ(WML)、または前述の開示に定義されるマルチモーダル/会話型ブラウザへのアクセスを提供したい場合に、アプリケーション・プログラマにいくつかの問題を提起する。第1に、この手法によれば、アプリケーション・プログラマは、アプリケーションを開発するときに少なくとも3つの異なる言語、例えばHTML、WML、およびVoiceXMLを扱わなければならない。すなわち、ユーザが従来の電話を介したスピーチ・ブラウザを介して、あるいはWAPブラウザを使用するまたは従来のウェブ・ブラウザを使用する無線接続を介してインターネット・ベースの情報にアクセスしようとするため、アプリケーションを書くときにHTML、WAP、およびVoiceXMLを使用しなければならないという事実を、アプリケーションは考慮しなければならない。これは、アプリケーション開発者にとって非常に厄介なものとして知られている。第2に、この手法によれば、マルチモーダル・アプリケーション、例えばアプリケーションにアクセスするために利用されるブラウザとの視覚とスピーチの両方に基づくユーザ対話を可能にするアプリケーションを同期させるのに適した方法がない。

【0007】アプリケーションは、従来、コンテンツ(すなわち情報またはその他のデータ)とプレゼンテーション(すなわちコンテンツがユーザに呈示される方式)の両方が混合されるように開発されてきた。しかし、アプリケーション・プログラミングを簡単にする試みにおいて、コンテンツをプレゼンテーションから分離する努力がなされてきた。こうしてXSL(Extensible Stylesheet Language)が開発されたが、これは、アプリケーションに関連するコンテンツがXMLで記憶され、特定の装置上でコンテンツを呈示するのに必要な変換がXSLによって扱われるように、XMLと共に動作する。<http://www.w3.org/Style/XSL>を参照されたい。このような手法は、W3C(ワールド・ワイド・ウェブ・コンソーシアム)によって採用されている。この手法は、メイン・ブラウザ(例えば異なるバージョンのMicrosoft Internet Explorer、Netscape Communicator/Nav

igator、その他のより一般的でないブラウザなど)の特徴にプレゼンテーションを適合させるのに通常使用される。この使用を他のモーダル性/チャンネル(例えば、組込み装置(無線電話またはPDA)の最上部にあるWMLのようなフォーマットをサポートする無線ブラウザ)に拡張しようとする者もいた。この最後の手法はあまり成功しておらず、または便利になっておらず、いずれの場合でも、これは、XSLページを複数回オーサリングすること必要とする。しかし、この手法は、アプリケーションと装置/チャンネルの両方に依存するという欠点を有する。すなわち、XSL規則は、コンテンツを転記すべきアプリケーションおよび装置に依存する。したがって、アプリケーションが新しい装置からアクセスされる場合、その装置用に新しいXSL変換を書かなければならない。

【0008】これらの問題のいくつかを克服するための他の試みもなされてきた。ユーザの意図(複雑かつタスク指向の意図)に基づいたXMLモデルを提供する試みがあった。ユーザの意図は複雑なコンポーネントでモデリングすることができるが、これは、小さなスクリーンまたはスピーチで装置上にレンダリングすることができないかまたは非常に難しい。それより小さい原子コンポーネントに分解されないこれらの複雑なコンポーネントはまた、モーダル性にまたがって密に同期させることができない。異なるブラウザによってレンダリングされる、装置独立のタグが提供される。また、スピーチ自動音声応答(IVR)システムへのいくつかの拡張が提案されてきた。しかし、欠陥の中でもとりわけ、これらの試みはダイアログをモデリングせず、モーダル性からモーダル性へのトランスコーディングは一般に不可能な作業である。

【0009】これらの手法では、ユーザの意図は、複雑な対話を記述する複雑なコンポーネントでモデリングされる。しかし、これらは通常、アプリケーション特定である。すなわち、これらは、ビジネス論理の概念および要素に依存し、またはそれを特徴づけ、または必要とする。したがってその場合、XSL規則(およびXSLスタイル・シート)が今日、根本的にアプリケーションまたはアプリケーション領域(すなわち関係するXML属性の性質)の関数であるのと同じ形で、これらの言語で書かれたページを変換するのに使用されるXSL規則もまた、根本的にアプリケーションまたはアプリケーション領域の関数である。これらは、新しいアプリケーションごと書き直さなければならない。これは、これらの手法の限界を特徴づけている。これらの手法は、アクセスのモーダル性に対して独立したコンテンツへのアクセスを提供するのを助ける際に貢献しない。実に、これらの手法は、そのアプリケーションまたはアプリケーション領域に關係するコンテンツへのアクセスを可能にするだけである。他のどんな場合も、変換規則の書直しを必

要とする。したがって、バックエンド・アプリケーションから変換規則を解放し、それをアクセス装置またはチャンネルによってサポートされる特徴/モーダル性だけに依存させることが必要とされている。

【0010】いくつかの場合に、複数のチャンネルのサポートは、スタイル・シートのカスケードを使用して、得られたXMLストリームをシリアル化された内部API(Application Programming Interface)として扱うことによって達成されてきたことに留意されたい。この場合もやはり、これは複数のオーサリングを必要とする。

【0011】さらに、上記の手法は、WMLのようなモーダル性で適切な対応するレンダリングを有しないコンポーネントによる、非常に複雑な意図モデルを有することになる。これらのモデルが、異なるタイプの(本質的に同じチャンネルおよびモーダル性の変形内の)表示装置またはブラウザの要件にグラフィカル・ユーザ・インタフェース(GUI)プレゼンテーションをカスタマイズする能力を提供するために設計されたことは明らかである。この結果、これらの手法はどれも、スピーチまたはマルチモーダルのユーザ・インタフェースを適切にモデリングせず処理しない。

【0012】すでに述べたように、従来のトランスコーディング(XMLコンテンツを呈示するのに使用されるXSL規則、およびあるモーダル性から別のモーダル性に移るためのXSLスタイル・シートの変更)は、異なるアクセス・モーダル性をサポートすると考えられてきた。これは、所与のXMLコンテンツに対し、システムが、XML規則を変更することによってHTMLページ、WML規則、またはVoiceXMLページやその他さえも作成することができることを意味する。実際、これは今日、市販の異なるウェブ・ブラウザ、例えばNetscape Communicator、Microsoft Internet Explorer、Sun Microsystems Hot Java、Spyglassブラウザ、Open Source Amayaブラウザ/エディタなどをサポートするのに使用されているものである。残念ながらこれは、以下の場合のみに可能である。

(i) XSL規則がアプリケーションまたはアプリケーション領域(すなわちXML属性の性質)に特定であり、

(ii) トランスコーディングが2つの言語間、例えばHTMLとWMLとの間であり、元のコンテンツが非常に厳格なオーサリングの規則を守りながらHTMLで構築された場合。もっとも、これは、所与の会社内の所与のウェブ・サイトに対する場合にのみ施行可能である。そのような場合でも、これは、一般にはほとんど実施不可能である。というのは、他のモーダル性で対応するコンポーネントを提供するための、マークアップ言語またはモーダル性にまたがる情報の欠落(例えば、HTMLフォームまたはメニューが音声によって自動的にそれをレンダリングするために必要な情報を提供しない)、なら

びに異なるモーダル性におけるダイアログ・ナビゲーション・フローの違いのせいである。

【0013】したがって、既存の言語およびブラウザにあるとされるこれらおよび他の欠点を克服する、アプリケーション・プログラミング言語およびそれに関連する情報ブラウジング機構が必要とされている。

【0014】

【発明が解決しようとする課題】本発明は、ユーザが任意のタイプの情報にアクセスするために利用している任意の装置とのユーザ対話に基づく新しいアプリケーション・プログラミング言語を提供する。この新しい言語を、本明細書では「会話型マークアップ言語 (Conversational Markup Language (CML))」と呼ぶ。

【0015】

【課題を解決するための手段】好ましい一実施形態では、CMLは、ユーザが所与のコンピューティング装置と行うことになる「ダイアログ」または「会話」を表すためのハイレベルXMLベースの言語である。ダイアログおよび会話という用語を本明細書で使用するが、これらはより一般的に、モーダル性および装置に対して独立した、装置（ローカル装置、リモート装置（例えば電話を介した対話）、または他の方法で分散されたいずれかの装置）とのユーザ対話を指すことを理解されたい。したがって対話は、これらに限定されないが、視覚ベース（テキストまたはグラフィカル）のユーザ対話およびスピーチ・ベースのユーザ対話、ならびにこれらの組合せを含むことができる。

【0016】このような言語により、アプリケーション作成者は、本明細書で以後「会話型ジェスチャ (conversational gesture)」と呼ぶ対話ベースの要素を使用してアプリケーションをプログラムすることができる。会話型ジェスチャは、それに従ってプログラムされたアプリケーションに関連する情報にアクセスするために利用されるモーダル性、装置、またはブラウザに対して独立してどんなダイアログも記述する、CMLの基本的なプログラミング・コンポーネントまたは要素である。

【0017】本発明は、新しいアプリケーション・プログラミング・パラダイムを提起することにより、これらおよび他の特徴および利点を達成する。前述のように、既存のアプリケーション・オーサリング手法は、アプリケーションのコンテンツ・ベースのアスペクトをプレゼンテーション・ベースのアスペクトから分離する概念を採用してきた。本発明によれば、CMLは、アプリケーション・プログラミングをコンテンツ・アスペクト、プレゼンテーション・アスペクト、および対話アスペクトに分離することを可能にする新しいパラダイムを導入する。ユーザに関してアプリケーションの対話アスペクトに焦点を当てることにより、アプリケーションは、コンテンツ／アプリケーションの論理およびプレゼンテーションに対して独立した形で書くことができる。アプリケ

ーションのコンテンツ論理またはビジネス論理、あるいはその両方は、アプリケーションに関連する「バックエンド論理」とも呼ぶことを理解されたい。

【0018】クライアント／サーバ構成では、「バックエンド論理」は、論理すなわち、アプリケーションの進化を促す状態および状況のコード化されたセット、ならびに可変の妥当性検査情報を含むアプリケーションの部分である。追って説明するが、バックエンド・データから分離された論理情報を伝えるために、属性制約および妥当性検査情報をCMLページに加えることができる。したがって、以下に説明し例示するが、アプリケーションがCMLで作成された後、そのアプリケーションに関連するCMLコードの一部がサーバからクライアント装置にダウンロードされ、次いで、CMLコードのCMLジェスチャが、その装置で利用されるブラウザ特定のマークアップ言語、例えばHTMLまたはVoiceXML、あるいはその両方にトランスコーディングされる。

【0019】本発明によれば、ダウンロードされたCMLコードで動作する装置（クライアント、さらにはCMLページをHTML、VoiceXML、WMLなど、おそらく他のレガシー・マークアップ言語にサービスするサーバ）は、例えばHTMLおよびVoiceXMLにほぼ同時にトランスコーディングし、したがってユーザに情報へのアクセスを提供する複数のブラウザを同期させることができる。本発明によるこのような有利な同期化は、トランスコーディングがジェスチャ識別によってジェスチャごとに行われるために可能である。したがって、あるモーダル性で入力／出力イベントが発生するとき、ブラウザは、どんなイベントがどんなジェスチャに対して発生したかを知り、サポートされるすべてのモーダル性を即座に更新することができる。これにより、モーダル性にまたがる非常に密な同期化がもたらされる。このような同期化はまた、例えばグラフィカル・ユーザ・インタフェース (GUI) ブラウザまたはスピーチ・ブラウザに関連する、モーダル性特定の様々なユーザ・インタフェース・ダイアログが、ジェスチャごとに単一のCML表現から生成されるという事実によっても達成される。したがって、ユーザが一方または他方のモーダル性を対話的に続行するのに伴い、複数のユーザ・インタフェース、例えばGUI、スピーチなどが同期化され、連続的に更新される。本発明のCMLおよびブラウジング機構は、自然言語 (NL) プログラミング用のプラットフォームも提供する。CMLによってアプリケーション作成者がジェスチャごとにプログラムすることができるため、そのようなアプリケーションは、広範囲の自然会話方式で要求／応答を提供するフレキシビリティをユーザに提供する。したがってユーザは、単一のコマンドに制約されるのではなく、より制約されない形で、例えばより自然の会話に近い形で、アプリケーションと対話することができる。NLおよび本発明によって、ユーザ

は、もう1人の人間と行っているかのような自然な会話を行う以外にどんな制約もなく、複数のモーダル性で自由に自身を表現することができる。NLの場合にはさらに、システムは、コンテキストおよび過去の対話／ダイアログ履歴（ならびにユーザ・プリファレンス、アプリケーション設定、記憶された共通知識などの他のメタ情報）を使用して、照会を一義化することができる。

【0020】NLは、スピーチに限定されず、自然マルチモーダル会話型アプリケーションのすべてのアспектを含むステートメントである。これは、NL入力を自然マルチモーダル入力と結合する。上に参照した整理番号Y0999-111によって識別される特許出願に記載のように、どんな入力も、モーダル性に対して独立して入力／出力イベントとしてモデリングされ、次いでそれは、ダイアログ・マネージャおよびアービトラータによって処理され、ダイアログ・マネージャおよびアービトラータは、履歴、ダイアログ・コンテキスト、および他のメタ情報（例えばユーザ・プリファレンス、装置およびアプリケーションの情報）を使用して、入力イベントのターゲットを決定する、あるいは要求されたアクションを実行する前にダイアログをユーザに関係させてユーザの意図を完結、確認、訂正、または一義化する、あるいはその両方を行うことになる。

【0021】本発明がマルチデバイスまたは分散ブラウジング環境を可能にすることも理解されたい。複数のブラウザを効果的に同期させるCMLの性質およびその能力により、アプリケーションの様々な部分を別々のコンピューティング・デバイス上に常駐させて実行することができる。次いでユーザは、あるアプリケーションにアクセスするときに複数の装置、例えばラップトップ・コンピュータおよびセルラー・ホンと同時に対話することができる。これを「マルチデバイス・ブラウジング」と呼ぶ。実際、本発明のこの態様は、「マルチモーダル性」を必要としない。すなわち、GUI／HTMLブラウザだけでも、ジェスチャ・ベースのXSL規則を使用して、どのブラウザに何がレンダリングされるかを定義することができる。したがって、いくつかのコンテンツを携帯情報端末（personal digital assistant）すなわちPDA上に表示し（すなわちカラー画像、ストーリーミングされたビデオ、長いリスト）、残りのコンテンツをセルラー・ホン画面上などに表示することができる。

【0022】CMLがモーダル性独立なので、アプリケーションが書かれた後でも、どんなタイプのブラウザに関連するどんなトランスコーディング規則も実施することができる。すなわち、CMLにより、作成者は、元々実装されていた可能性のあるデフォルトのトランスコーディング以外の別のタイプのトランスコーディング（すなわちジェスチャ・ベースのトランスコーディング規則）に変更することができる。したがって、この特徴は有利にも、ジェスチャ・ベースのXSL規則の単純な更

新によって、いわゆる「レガシー言語」、例えばHTML、WML、VoiceXMLなどの新しいリリース／バージョンと、新しい言語、例えばCHTML、HDMLなどに対するサポートを保証する。さらに、この特徴は、単純なジェスチャ・ベースのXSL規則を使用して、あるバージョンのCMLから新しいCMLへの単純かつ容易な通路を可能にする。バージョンからバージョンへのジェスチャごとのトランスコーディングは、CMLから他のレガシー言語へのトランスコーディングと異なる問題ではないことを理解されたい。これは、CMLが定義によってこのトランスコーディングの原理を取り巻くように設計されるので、特に有利である。これが他のマークアップ言語のほとんどに当てはまらないことは確かであり、他のマークアップ言語では、後方互換性は提供するかもしれないが、仕様の更新は通常、新世代ブラウザに対して、ならびにより古いバージョンで書かれたより古いすべてのコンテンツに関して問題を含む。

【0023】CMLはまた、CMLページが書かれた後でも、プレゼンテーションを表面的に変更することを可能にする。例えば、所望のモーダル性およびターゲット・マークアップ言語に応じてCMLコマンドを発行して、いくつかのモーダル性におけるコンテンツのプレゼンテーションの、いくつかのフィーチャを表面的に変更することができる。これにより、CML開発者は、HTMLレンダリングに対してするのと同じ量の表面的努力で済む。しかし、利点は当然、（アクセス装置またはチャネルに対して独立）ユニバーサル・アクセスを提供するのに使用できる、またはマルチモーダルおよび会話型ユーザ・インタフェースを密に同期させることができる、あるいはその両方ができる、対話のマルチチャネル（すなわち複数のタイプのターゲットMLまたはデバイス・モーダル性または特定のユーザ・インタフェース特徴で表現することができる）記述を、同じ代償で得たことである。

【0024】本発明はまた、様々なモーダル性特定の表現、例えばHTMLベースのグラフィカル・ユーザ・インタフェース（GUI）ブラウザ、VoiceXMLベースのスピーチ・ブラウザなどに従ってCMLの機能をサポートすることのできるマルチモーダル・ブラウザの様々な実施形態を可能にする。

【0025】用語「CML」は、上で参照した整理番号Y0998-392およびY0999-178によって識別される特許出願で使用されていることに留意されたい。これらの出願では、この用語は、会話型インタフェースを記述するための宣言的な方式を指すものとされる。本発明によれば、用語CMLは、以下に詳細に述べるように、対話によるプログラミングの概念を組み入れたジェスチャ・ベースの言語について言う。

【0026】本発明のこのような態様ならびに以下に述べる他の態様があった場合に、ここで、このような創意



に富んだ特徴と既存の手法との重要な違いをいくつか考察する。この5年間のワールド・ワイド・ウェブ(WWW)の急激な成長は、まずコンテンツからユーザ対話を分離し、続いてプラットフォームに依存するWWWブラウザによってレンダリングされるHTMLのようなマークアップ言語を介してアプリケーション・フロント・エンドを送達することによって軽量ユーザ・インタフェース・アプリケーションを構築することにおける潜在的な強さを指摘した。この体系は、基礎をなすハードウェアおよびオペレーティング・システムの細部からエンドユーザ・アプリケーションを解放することによって、新しい可能性の世界を開く。現在のWWW体系は、基礎をなすハードウェアおよびオペレーティング・システムの細部から、eコマース・アプリケーションへの視覚インタフェースを開放した。この進化における次のステップは、インタフェース・モーダル性と、電子情報と対話するのに使用される装置とに対して独立したエンドユーザ・アプリケーションを作ることである。この進化は、新世代のeコマース・アプリケーションとのスピーチ・ベースの対話を可能にすることにおいて自然な次のステップである。

【0027】装置およびモーダル性から独立したエンドユーザWWWサービスを達成するために、様々な装置への送達を可能にするような、モーダル性独立の技術を使用するアプリケーションおよびサービスをオーサリングすることが強く必要とされている。XMLが急速にWWWの次世代共通語となっていることから、そのような言語をXMLアプリケーションとして設計することが自然である。

【0028】したがって、モーダル性独立の情報コンテンツおよび対話論理をオーサリングするためのXMLベースの言語を設計し、次いで、得られたアプリケーションをターゲット装置に最適な仕方では送達することにより、モーダル性独立のWWWサービスを達成することができる。これは必然的に、情報コンテンツ、情報プレゼンテーション、および対話論理を別個のコンポーネントに分離する言語を設計することになる。WWWはすでに、スタイル・シートを利用することによってコンテンツをプレゼンテーションから分離することに向けて進化している。次の進化ステップは、対話論理を情報コンテ

<prior art ML>

<CHOICE  
SELECTION-POLICY="SINGLE">

<CAPTION>Title</CAPTION>

<HINT>This is a set of valid titles for a person.</

HINT>

<STRING NAME="Mr">

<VALUE>Mr.<VALUE>

</STRING>

<STRING NAME="MRS">

ンツから抽出することである。現在、この領域での外部規格活動が、W3Cなどの産業コンソーシアムのXFORMSおよびボイス・ブラウザの委員会内から現れることが予想されている。

【0029】上に概説した分離は、会話型コンピューティングと我々が呼ぶ手法となる。エンドユーザ・アプリケーションおよびサービスは、モーダル性独立の会話型ジェスチャの集合として表され、各会話型ジェスチャは、ユーザ対話を構成するマン・マシン・ダイアログの原子的な一片をコード化する。

【0030】上に概説した洞察は、意図ベースのマークアップ言語を設計する試みが近い過去にわずかにあったという事実によって妥当性検査される。これらは最初に、異なる装置間、例えば小さい画面のハンドヘルド対デスクトップPCの視覚プレゼンテーションにおける違いを抽出するように設計された。スピーチ・インタフェースが関係してくるとき、これらの言語は両方とも、これらの元のターゲットであった異なる視覚表示に加えてスピーチ装置に送達するためのエンドユーザ・アプリケーションをオーサリングするための可能な手段として呈示される。

【0031】本発明によれば、CMLは、スピーチおよび自然言語技術を含む会話型インタフェースに対処する必要性によって導入される新たな要件に特に焦点を当てられて、モーダル性独立のユーザ対話に向けたXMLベースの言語として最初から設計される。ユーザ・インタフェースにおける一流市民としてのスピーチへのこの焦点は、CMLを従来の試みとは異なる形で進化させた。これらの、鍵となる違いのいくつかを対比することにする。

【0032】(i) データ・モデル上への対話のオーバーレー

従来技術の言語はすべて、ユーザの意図と、ユーザ対話によって同じ一片のマークアップ内に配置される、基礎をなすデータ・モデルとを定義する。ある仕様からこれを例示する短い例を挙げる。以下に示すマークアップの断片は、人の称号(Mr.、Mrs.、Ms)を得るのに使用されることになる。プロンプト指示されるデータ・モデルの定義が、ユーザ対話を生成するマークアップと混ざっていることに留意されたい。

```

<VALUE>Mrs.</VALUE>
</STRING>
<STRING NAME="MISS">
  <VALUE>Miss</VALUE>
</STRING>
<STRING NAME="MS">
  <VALUE>Ms</VALUE>
</STRING>
</CHOICE>

```

</prior art ML>

【0033】上記を、以下に示す、人の称号を得るためのCML表現と比較されたい。データ・モデルの定義、すなわち有効な人の称号をリストする列挙型を、ユーザ対話コンポーネントすなわち選択ジェスチャから分離していることに留意されたい。

【0034】まず、列挙型PersonTitleを定義する。  
 <enum name="PersonTitle" type="string"> <value>MR</value> <value>MRS</value> <value>MISS</value> </enum>

【0035】フィールドPersonTitleが定義されると、それは、ユーザ対話中の複数のポイントで適切なCMLジェスチャを介してインスタンス化することができる。これを、ジェスチャselectによって以下に示す。

```

<select name="PersonTitle" selection-policy="single"> <message>PersonTitle</message> <choices> <choice value="MR">Mr.</choice> <choice value="MRS">Mrs.</choice> <choice value="MISS">Miss.</choice> </choices> </select>

```

【0036】基礎をなすデータ・モデルの定義（上の列挙PersonTitle）から会話型ジェスチャ（上の例ではジェスチャselect）を分離することは、次のようないくつかの利点をもたらす。

【0037】（1）データ定義から会話型ジェスチャを分離することにより、例えば上のダイアログを国際化するとき、人の称号をプロンプト指示するための複数のユーザ・インタフェースをオーサリングすることができる。したがって、CMLで構成されたこのダイアログのドイツ語バージョンは、会話型ジェスチャを修正されることだけしか必要としないことになる。上記の表現が国際化されるとき、すなわち変更が必要なのは要素caption、hint、およびcode valueのコンテンツであるとき、基礎をなす列挙型の定義は同じままであることに留意されたい。しかし、ユーザ・インタフェース・マークアップをデータ定義上にオーバーレーすることによれば、この設計は、ダイアログを国際化するのに必要な変更を切り離すことができない。従来の言語のいくつかは、上記のダイアログの異なる言語バージョンを作成するために作成者によって次いで再利用されるテンプレートの概念を導入することによって、国際化に関するこの明示的な問題を回避していることに留意されたい。しかしこれ

は、根源にある基本的な問題を除去していない。すなわち、データ定義およびユーザ・インタフェースは、依然としてテンプレート定義中でリンクされたままである。

【0038】（2）フィールドPersonTitleが定義されると、CMLジェスチャは、ユーザ対話中の複数のポイントでこのフィールドを参照することができる。したがって、ユーザがフィールドPersonTitleに対する値を指定すると、ダイアログの後続の部分は、プロンプト、例えばWelcome to the electronic store Mr. Smithを作成するときに、供給された値を参照することができる。

【0039】（3）CMLでオーサリングされたアプリケーションはまた、ユーザ対話の異なるポイントでPersonTitleなどの特定のフィールドに対してユーザに自由にプロンプト指示することができ、ユーザは、どのポイントで自分がそのフィールドに値を供給するかを自由に決定することができる。このような形のフレキシビリティは、自然言語インタフェースの設計では特に不可欠であり、この場合もまた、ユーザ対話を宣言するマークアップからモデルを定義するマークアップを分離する結果となる。この分離がなければ（現時点の従来技術のように）、上記のものは、作成者にフィールドPersonTitleを複数回定義させることになる。

【0040】上記のことを理解するために、ユーザがミューチュアル・ファンドを売買することができると同様に特定の資産の正味価値を知ることでもできるミューチュアル・ファンド・アプリケーションを考察されたい。この対話を単純化したバージョンで、システムは、ユーザから2つの情報アイテムを得る必要がある。

（a）ユーザ・アクション、すなわち売買または正味資産評価

（b）作用する資産、例えば購入資金

【0041】上の例に自然言語インタフェースを使用するとき、ユーザは、システムから最初にプロンプト指示されるときに、実行するアクションと作用する資産のいずれか、またはおそらくその両方を指定することが等しく見込まれる。何が指定されたかに応じて、次にダイアログは、欠落した部分の情報に対してシステムがプロンプト指示する状態に移行する必要がある。あるいは、アクションと資産が両方とも指定された場合、システムは、フォーム「Would you like to action specified f



und?」の確認プロンプトを作成する必要がある。従来技術は現在、対話マークアップすなわちこの場合に要素CHOICEをデータ定義上にオーバーレーするので、アプリケーション作成者が同じフィールドの値、例えばユーザ対話中の異なるポイントにおける資産を得るためにユーザ対話を指定することは不可能となる。

【0042】データ・モデルの上への対話のオーバーレーは、本明細書に開示する、我々の手法の新規性とプログラミング・モデルの新しいパラダイムとを特に強調する。

【0043】(ii) アプリケーション状態をカプセル化するための明示的環境の欠如

CMLでユーザ対話からデータ・モデルを分離することの別の結果は、CML文書としてオーサリングされたアプリケーションが、アプリケーション状態、例えば上に挙げた例の中のPersonTitleまたはアクションを結合する環境を明瞭に呈示することである。従来技術の場合では、このアプリケーション状態は暗黙的であり、その言語でコード化されたユーザ・インタフェースの他の部分にとって容易に利用可能ではない。

【0044】データ・モデルを、したがってアプリケーション状態を明示的に定義することにより、CMLは、ユーザ対話が完了するとサーバに送り返されることになるXMLコード化を明瞭に定義する。したがって、フィールドPersonTitleの場合、サーバは、サブミット時に以下のものを受信することになる。

<PersonTitle>MR</PersonTitle>

【0045】データ・モデルの定義にアクセスできるサーバは、サブミットされた値を妥当性検査することができる。より複雑な例では、データ・モデル定義は、アプリケーション特定の妥当性検査制約をカプセル化することができ、これらの制約は両方で、クライアント側でチェックして、後でサブミット時にサーバ・エンド上で妥当性検査することができる。このようにデータ・モデルおよび制約をユーザ・インタフェースから分離することにより、ユーザが特定の対話装置、例えばデスクトップPCを使用して対話を開始することを可能にするCMLアプリケーションが、部分的に完了したトランザクションをサブミットし、後で異なる装置、例えばセルラー・ホンを使用してそのトランザクションを完了させることができる。

【0046】(iii) 従来技術はGUIレガシーを反映する

従来技術の仕様に定義されているコア属性の多くは、GUI特定のレガシーを反映している。例えば、すべてのデータ・モデルは表示されるコア属性によって制限され、これは、表示ベースのインタフェースにしか意味をなさない。スピーチ・ベースのハンドヘルドやセルラー電話などの視覚的でない装置に対し、enable=false、shown=trueなどの一義的な設定変換はないように思われ

る。

【0047】さらに、これらの属性は、ユーザ対話の表現を小さいサイズの表示装置にマッピングすることを難しくする。これは、デスクトップGUI用のこれらのMLでオーサリングされるアプリケーションが、対話要素の多くを表示されるように宣言する見込みが高いからであり、これは表示装置のスペースが乏しい環境では難しくなる。

【0048】従来技術は通常、大きな画面の外では意味を持たない他のGUIコンポーネントを有する。残念ながら、言語内に行き渡り、モーダル性／チャンネルにまたがって容易に使用可能ではない機能には問題があり、トランスコーディング／レンダリングが任意のターゲットに対して可能となることを保証することができない。

【0049】さらに、スピーチのようなモーダル性は、ダイアログ・コンポーネントをレンダリングするために追加の情報（例えば文法、語彙、言語モデル、音響モデル、NL解析およびタグ付けデータ・ファイルなど）を必要とする可能性がある。この情報は、従来の仕掛けでは利用不可能である。この場合もまた、データ・モデルと対話の間のオーバーレーは、同じダイアログ・コンポーネントが、異なるデータ・ファイルを有するページで複数回使用されるときに問題となる。

【0050】(iv) 原子的会話型ジェスチャの欠如  
従来技術によるユーザ対話の表現は、実装されている、基礎をなすデータ・モデル上に直接オーバーレーされるため、これらのMLには、CMLにあるような原子的会話型ジェスチャのセットの概念はない。むしろ、selectなどの明示的なCMLジェスチャは、従来技術では暗黙的である。例えば、CMLジェスチャselectは、従来技術では、リスト構造に対するマークアップ上に選択要素に対するマークアップをオーバーレーした結果として表れることになる。上に挙げたフィールドPersonTitleの例を参照されたい。

【0051】より複雑なダイアログを構成するとき、原子的会話型ジェスチャの欠如がまず問題になる。例えば、従来技術は、明示的なテーブルおよびツリー構成を導入して、2次元表レイアウトおよびツリーの仕組みのGUI概念に対応する。しかし、これらのよりハイレベルな構造はCMLにおけるように原子構成単位で構築されていないため、テーブルやツリー（ここでツリーはopenまたはclosedと宣言される）のようなコンポーネント構成を、静的2次元表示が欠如したスピーチのようなモーダル性にマッピングすることは不可能である。また、ツリーやテーブルのようなジェスチャには、画面の小さい装置上で直接に相当するものがない。

【0052】(v) 同期化  
複数の対話モーダル性にまたがる密な同期化が、高品質なマルチモーダル・インタフェースの鍵となる要件である。さらに言えば、このようなマルチモーダル・クライ

アントは、基礎をなすプラットフォームとして従来のブラウザから提供されるDOM (Document Object Model、<http://www.w3c.org>に記載)を使用して構成される見込みがより高い。この後者の実施シナリオでは、上で詳述した、ユーザ・インタフェース構成をデータ定義上にオーバーレーすることが、やはりネックになりやすい(例えば、ダイアログ/アプリケーション状態をカプセル化するための明示的な環境の欠如に対して上に述べた、この場合にビューごとの同じ問題)

【0053】モダリティにまたがる密な同期化は、CMLにおける基本的な目標である。これは、CML設計全体に反映され、会話型ジェスチャとデータ・モデルの定義との間に生じる分離は、旧来のモデル・ビュー・コントローラ(MVC)設計を使用してDOMの最上部に構築されるマルチモーダル・ブラウザの実装をより容易にする。

【0054】(vi) 会話型アプリケーション  
会話型アプリケーションは、複数のフォーム(それぞれがトランザクションまたはトランザクションの一部を記述する)を同時に活動化させることによって宣言的に開発することができる。これは、ファイル中の異なる場所で同じダイアログ・コンポーネントを再利用する機能を必要とする。上に説明したように、前述のオーバーレーはこの要件をサポートしない。

【0055】(vii) イベント結合の欠如  
イベント結合能力の欠如は、アプリケーションのマルチチャネル/マルチモーダル機能を制限する。すなわち、何らかの具体的な論理動作を何らかの具体的な物理動作に関連付ける方式がない。これは、異なる結合が望ましいマルチモーダル/マルチチャネル・アクセス(例えば電話ヘルプ用キー・ショート・カット、ヘルプ用音声コマンド、キーボード上におけるヘルプ用のキーの組合せ)を提供したい場合にクリティカルである。

【0056】(viii) ピア  
さらに、従来技術の試みはまた、基礎をなす同じ表現から異なるユーザ対話を生成するためにピアの技術に依拠している。そうすることにより、これは、同期化されるマルチモーダル対話の問題に対処しない。

【0057】  
【発明の実施の形態】後続の説明で、好ましい仕様のCML、好ましいマルチモーダル・ブラウジング環境、および本発明をよりよく理解するためのいくつかの例示的な適用例を使用して本発明を示す。しかし、本発明がこれらの特定の好ましい実施態様および例示的な適用例に限定されないことを理解されたい。本発明はむしろ、より一般に、アクセス・プロトコル、モダリティ、ブラウザ、または装置に関係なく、どんな情報アクセス・アプリケーションにも適用可能である。したがって、本発明はより一般に、同期化されマルチモーダルの、簡単かつ便利な情報のアクセスをユーザに提供することが望まし

いどんな情報アクセス状況にも適用可能である。

【0058】詳細な説明は、参照しやすいうに次のセクションに分かれている。すなわち、(I) CML仕様、および(II) CMLをサポートし、解析し、レンダリングするためのマルチモーダル・ブラウザ体系である。セクションIは、本発明によるCMLの好ましい仕様の詳細な説明を提供する。セクションIIは、本発明によるCMLを実施するための好ましいマルチモーダル・ブラウジング環境を提供する。

#### 【0059】I. CML仕様

以下の説明は、CMLの好ましい実施形態の仕様である。このセクションは、参照しやすいうに次のサブセクションに分かれている。すなわち、(A) 序説、(B) 比較例、(C) CML構文、(D) 名前空間、(E) CML属性、(F) CMLコンポーネント、(G) 結合イベント、(H) ジェスチャのグループ化および定義フォーカス、(I) データ・モデルおよびデータ・モデル、(J) アクセス環境、(K) CML横断モデル、(L) 特定ユーザ・インタフェース言語へのCMLの変換、(M) 表面変更、および(N) CML文書型定義である。

#### 【0060】A. 序説

前述のように、コンテンツ再利用を達成するためにコンテンツをプレゼンテーションから分離することは、ワールド・ワイド・ウェブ(WWW)上で情報を展開するために従来受け入れられてきた方式である。これを図1に示す。図示のように、アプリケーション・オーサリングに関する既存の手法は、2つのコンポーネントしか考慮しない。すなわち、コンテンツ・コンポーネント(A)、およびプレゼンテーションコンポーネント(B)である。現在のW3C体系では、このような分離は、コンテンツをXMLで表現し、次いでそれをアプリケーションおよび装置に依存するXSL変換を介して適切な最終形式のプレゼンテーション(例えばHTML、VoiceXML、WML)に変換することによって達成される。しかし、この手法にはクリティカルな欠点がある。実際、XSL規則は通常、バックエンド・アプリケーションまたは領域に依存する。結果として、アプリケーションのオーサリングは、XMLコンテンツを設計し、次いでXSLスタイル・シートをアプリケーション/ページごとに、かつターゲット装置/チャネルごとに設計することを伴う複数のオーサリング実行である。さらに、スタイル・シートを使用してあるMLから別のMLにトランスコーディングされると予想されるとき、前述のように、トランスコーディングは通常、2つのレガシー言語間(例えばHTMLからWMLへ)であることが多く、次いで、非常に厳密なオーサリングの規則に従って元のコンテンツがHTMLで構築される。もっとも、これは、所与の会社内の所与のウェブ・サイトに対する場合にのみ施行可能である。そのような場合でも、

これは、一般にはほとんど実施不可能である。というのは、他のモーダル性で対応するコンポーネントを提供するための、マークアップ言語またはモーダル性の間の情報の欠落（例えば、HTMLフォームまたはメニューが音声によって自動的にそれをレンダリングするために必要な情報を提供しない）のせいである。

【0061】CMLは、フォーム（プレゼンテーション）およびコンテンツに加えて第3のコンポーネント、すなわち対話を実現することによって動かされ、この対話は、静的情報表現を対話型情報に変えることの中心にある。静的情報は、ユーザが受動的であり、かつ、すべての情報を呈示される、非常に特殊な場合であることを理解されたい。この新しいパラダイムを図2に示す。図示のように、本発明は、対話によるプログラミングの概念を導入し、ここで、アプリケーション・オーサリングは3つのコンポーネント、すなわちコンテンツ（A）、プレゼンテーション（B）、および対話（C）に分割される。この新しいプログラミング・パラダイムは、新しいプログラミング環境、例えば開発ツールなどの開発と組になって行われる。

【0062】この明細書を通して、少量の埋め込み型アプリケーション・インテリジェンスを有する「軽量」情報アプリケーションまたは電子情報を「インフォウェア（infoware）」と呼ぶ。現在まで、このような対話は、一部は表示的なHTMLすなわちフォーム要素内で、一部はサーブレット（servlets）およびCGI（CommonGate Interface）スクリプトにカプセル化されたサーバ側の論理内で表示されてきた。この組合せが、情報コンテンツが支配するインフォウェアすなわち軽量アプリケーションを生み出すことになった。今日のWWWにおけるインフォウェアのよい例は、Amazon.comのようなeビジネスである。

【0063】複数のモーダル性を介してこのようなインフォウェアと対話する世界に我々が移動するのに伴い、この3つのアспект、電子コンテンツすなわちコンテンツとプレゼンテーションと対話との間を明瞭に分離する時が来た。

【0064】CMLは、すべてのマン・マシン・ダイアログが、適切なシーケンスの「会話型ジェスチャ」、または適切に結合させて任意の対話に置き換えることのできるモーダル性独立の構成単位（コンポーネントまたは要素）に分割できるという洞察に基づく。CMLは、これらの基本的な構成単位をXMLでコード化することにより、モーダル性独立方式でマン・マシン対話をカプセル化する。このようなCMLカプセル化は、後に適切なモーダル性依存のユーザ・インタフェースに変換される。この変換は、複数の「コントローラ」、すなわち今日のWWW中心の世界におけるブラウザの間で同期化を達成する形で行われ、これらが、単一のモーダル性独立「モデル」に対するモーダル性特定の「ビュー」を操作

する。「モデル」、「ビュー」、「コントローラ」という用語は、コンピューティングの従来のMVC（モデル・ビュー・コントローラ）分解に従って使用される周知の用語であり、例えば、その開示を参照により本明細書に組み込むG.E.KrasnerおよびS.T.Popeの「A Cookbook for Using the Model-View-Controller User Interface Paradigm in SmallTalk-80」, Journal of Object-Oriented Programming, 1(3):26-49, August/September 1988を参照されたい。結果として、複数の情報機器にわたって対話挙動が一貫し、複数のインタフェース・モーダル性にわたってユーザ対話が整合され正しく同期化される。

#### 【0065】B. 比較例

CMLの好ましい実施形態の仕様を説明する前に、CMLおよび対話によるプログラミングの基本原則を示すいくつかの例を呈示する。これらの例は、「グローバル・カフェ」サイトに言及する。カフェに着く前またはカフェにいるときに顧客が自分の飲物を前もって注文することができるようにすると決定したカフェを想像されたい。したがって、彼らは基本的に、アクセス・チャネルに対して独立して自分の情報へのアクセスを提供したいと思う。

【0066】したがって、ページがCMLでオーサリングされる。このページを生成するためのCMLコードをCMLコード10として図3に示す。このページは基本的に、一連の会話型ジェスチャを含む（ここでジェスチャは、CMLおよび対話によるプログラミングの基本原則がよりよく理解されるように、後で提供される実際のCML仕様の細部からはいくぶん自由であることに留意されたい）。

【0067】このページは次のものを含むことができる。

- (1) タイトル（図3で「ジェスチャ」20として示す）：「Global Cafe」（すなわちタイトルとしてレンダリングされる特定のメッセージ）
- (2) ジェスチャ・メッセージ（図3で「ジェスチャ」22として示す）：Would you like coffee, tea, milk or nothing?
- (3) ジェスチャ・リストからの排他的選択（図3で「ジェスチャ」24として示す：リストは次のアイテムで構成される。coffee, tea, milk, nothing.
- (4) サブミット・ジェスチャ（図3には明示的に示さず）

【0068】明らかに、このページは、ターゲット・モーダル性（すなわちアクセス・チャネルまたはアクセス装置のタイプ）へのどんな依存も導入せずに、ユーザとの完全な対話を十分に定義している。このページはまた、次のような、対話によるプログラミングのプログラミング・モデルもはっきりと示している。

【0069】(i) ターゲット・モーダル性に対して独

立して対話の基本的なコンポーネントを使用して、アプリケーションが対話によってプログラムされる。

( a ) ジェスチャ・メッセージ : 「 Global Cafe 」

( b ) ジェスチャ・メッセージ : Would you like coffee, tea, milk or nothing?

( c ) ジェスチャ・リストからの排他的選択

( d ) サブミット・ジェスチャ

【 0070 】 ( ii ) これは、従来通りにプログラム／開発されるバックエンドに接続される。この例では、バックエンドへの接続はリスト ( coffee, tea, milk, nothing ) によって例示され、これは、ページが作成されたときには静的に、また、ページがバックエンド論理を使用してサーバ上で動的に生成されたときは動的に、バックエンド・データベース中に読み込まれ、かつリストへの引数として追加されたものである。

【 0071 】 ( iii ) この段階で必要なら、例えば XFORM 構文を使用して、属性／変数の制約、妥当性検査を追加することができる。例えば、ページがアルコール飲料を提供するためにユーザの年齢を要求する場合には、ユーザが自分は未成年であることを示すとダイアログを制約または修正するといった制約を容易に表すことができる。これは、このページには明示的に示していない。

【 0072 】 ( iv ) プレゼンテーションは以後、表面変更することができる。この例では、これは、ジェスチャ・メッセージの代わりにジェスチャ・タイトルを使用することによって行われる。すなわちモーダル性独立の表面変更である。モーダル性特定の表面変更もまた、例えば、得られた HTML ページに使用するための背景 ( 色または画像 ) を指定する HTML タグを追加することによって追加することができる。これは、他のターゲット・モーダル性からは無視される、あるいは他のモーダル性用に提供された「挙動」で置換されることになる。例えば、HTML モーダル性で画像が表示されるとき、代わりにレンダリングとなるキャプションを WML、VoiceXML、または他のモーダル性に提供することができる。

【 0073 】 ( v ) 得られたページは、次に、適切なブラウザによってレンダリングすることができる。2つのモデルが存在する。CML ページは、CML コンテンツを解析およびレンダリングすることのできるブラウザに提供される ( 以下のケース B 参照 ) か、またはレガシー言語、例えば HTML、WML、VoiceXML などしか扱うことのできないレガシー・ブラウザに提供される ( 以下のケース A 参照 ) 。

【 0074 】 ( a ) ケース A : このケースは、「マルチチャネル」ケースとも呼ばれる。ターゲット・ブラウザは、要求者のアドレス ( 無線ゲートウェイまたはスピーチ・ブラウザ ) から、または要求 ( すなわち HTML ファイル要求対 WML ページ要求 ) から明確である ( HT

ML ブラウザ用の HTTP 接続で識別される ) 。ページが要求されるとき、これは、CML で取り出され、ジェスチャ・ベースの XSL 変換規則を使用して実行中にターゲット ML にトランスコーディングされる。

【 0075 】 ( b ) ケース B : ターゲット・ブラウザは CML を扱う。したがって、ターゲット・ブラウザは、それがサポートするモーダル性は何であるか ( 単一か複数か ) 、ならびにサポートされるモーダル性で所与のジェスチャを最適にレンダリングするのに必要な規則を正確に知っている。これらのジェスチャ XSL 変換規則は、装置が構築されたときまたはブラウザがそれに移植されたときにブラウザ中にプログラムされたものであることが有利である。明らかに、これは、適切な情報を有する ( すなわちその装置を十分によく理解している ) 最も適切なプログラムがその責務を負うことを意味する。

【 0076 】 ( vi ) いずれの場合にも、CML アプリケーション開発者は、何もする必要がない。開発者は常に、プラットフォーム／ブラウザがレンダリングを適切に扱うと仮定することができる。

【 0077 】 ( vii ) ジェスチャは、ターゲット・モーダル性から完全に独立している。ジェスチャはまた、バックエンド・ビジネス論理／ドメインまたは他の何にも独立、ジェスチャだけに依存する。これが、XSL 規則をブラウザに記憶できる理由である。

【 0078 】 ( viii ) XSL 規則は、ターゲット・モーダル性に基いてジェスチャをレンダリングする。この場合では、これは次のことを意味する。

( a ) タイトル :

HTML : 太字の見出し文字が表示される

WML : 単一カード表示

VoiceXML : 歓迎プロンプト

( b ) メッセージ :

HTML : 通常の文字で表示

WML : 通常の文字で表示 ( おそらく複数のカード上で )

VoiceXML : プロンプト・メッセージを生成 ( テキストからメッセージへ、または再生 )

( c ) リストからの排他的選択 :

HTML : プルダウン・メニュー

WML : ラジオ・ボタン

VoiceXML : メニュー中で選択するためのダイアログ ( おそらく自然言語 ) ( 例えば、「あなたはどのように多くのアイテムから選べます。最初の3つを読みます。アイテムを選ぶか、次の3つが聞きたければもっと、と言ってください・・・」 )

【 0079 】 図3に戻ると、CML で書かれたグローバル・カフェ・アプリケーションで得ることのできる3つの例示的なレンダリングを視覚化したものがある。したがって、ジェスチャ・ベースの XSL 変換を含む CML コード 10 から、グローバル・カフェ・アプリケーション

ンのHTMLレンダリング12、WMLレンダリング14、およびVoiceXMLレンダリング16が得られる。

【0080】(ix) マルチモーダル／会話型ブラウザによってトランスコーディングが行われるとき（以下に述べる）、ジェスチャはnode\_idタグを使用して一意に識別される。これにより、登録された各モーダル性（ローカルまたは分散）でレンダリングが生成されるだけでなく、非常に密な同期化が実現する（すなわち、これが意味をなすジェスチャであるときに、ジェスチャ・レベルで、またはサブ・ジェスチャ・レベルでも）。例えば、イベント（I/Oイベント）は即座にダイアログの状態（すなわち、上に参照した整理番号Y0999178によって識別される特許出願にあるように、例えばマルチモーダル・シェル中に維持される状態）および他のモーダル性に影響する。したがって、このような密な同期化が、携帯情報端末がサポートできるHTMLレンダリング12と、従来の電話機がサポートできるVoiceXMLレンダリング16との間に存在することができる。

【0081】ジェスチャXSL変換規則は、アプリケーション開発者が上書きし、それらをどこでダウンロードすべきかを示すことができることに留意されたい。それらはまた、普通ならデフォルトの挙動となるものから、ユーザ、アプリケーション、または装置のプリファレンスによって上書きすることもできる。

【0082】新しいジェスチャを追加することもでき、その場合、関連するXSL規則を提供しなければならない（例えばそれを得る場所のURL）。

#### 【0083】C. CML構文

CMLの好ましい実施形態では、CML構文はXMLに準拠する。CMLインスタンスは、うまく形成されたXMLである。装置の制約に基づいてXMLプロセッサを妥当性検査するCMLプロセッサを実施することができる。

#### 【0084】(i) 特別なCML注記

##### (1) ケース依存性

CMLクライアントおよびサーバは、CML要素名および属性名をケース依存として扱う。慣習として、この仕様中に定義されたすべての要素名および属性名は、より低いケースを使用する。この慣習は、事前定義されたすべての要素名および属性名に厳密に課される。

##### 【0085】(2) コンテンツ・モデル

CMLインスタンスは、一連のXML要素からなる。CMLは、最上レベルにあるどんなp cデータも許容しない。すなわち、CMLのすべての最上レベルの子は、必ず要素である。

##### 【0086】(3) 乏しいCML

CMLインスタンスは乏しい場合がある。属性node\_idを除き、この仕様中に述べる最上レベルのCML属性および要素が必要とされる。

##### 【0087】(4) エンティティ参照

CML中のすべてのエンティティ参照は、URI (Universal Resource Identifier) 仕様に準拠する。http://www.w3.orgのW3CからURI仕様を参照されたい。

##### 【0088】(ii) 用語

CML文書を記述するのに使用される用語を、この仕様の本文中に定義する。以下のリスト中に定義する用語は、これらの定義を構築し、CML「プロセッサ」の動作を記述する際に使用される。CMLプロセッサは一般に、CMLコードおよび関連するアプリケーションを実行するように構成された処理装置を指す。用語は以下の通りである。することができる、してもよい(may)

適合したCML文書およびプロセッサは、記述されたように挙動することができるが、その必要はない。しなければならない(must) 適合したCML文書およびプロセッサは、記述されたように挙動する必要がある。そうでない場合は、以下に定義するエラーにある。エラー(error) 本仕様の規則の違反。結果は不定である。適合したソフトウェアがエラーを検出して報告することができ、それから復旧することができる。致命的誤り(fatal error) 適合したCMLプロセッサが検出してアプリケーションに報告しなければならないエラー。

##### 【0089】D. 名前空間

このセクションでは、CMLインスタンスのすべてのセクション内の名前空間の使用を詳述する。本仕様中に定義するすべての要素および属性は暗黙的に名前空間cml中にあることに留意されたい。すなわち、一般XML文書内で発生するCMLインスタンス中の要素名messageは、XMLプロセッサにはcml:messageとして見え、CML属性node\_idは、XMLプロセッサにはcml:node\_idとして見えることになる。このセクションの後続の段落で、CMLクライアント間の名前衝突を避けるために名前空間cmlがさらにどのように細分されるかを定義する。

【0090】「限定されない」名前空間によって導入されたすべての名前空間、例えばvxmlは、暗黙的に名前空間com.ibm.cml.vxml中にある。より一般には、ベンダ特定の名前空間は、ベンダのドメイン名から構成されるベンダ接頭部を使用する。これは、Javaのようなシステムによって使用される方式に類似する。

【0091】CMLはまた、名前空間を使用して、異なる個々のインフォウェアからのフィールド名および値が共存できるようにする。したがって、アプリケーションcafe中のフィールドdrinkの完全に限定された名前は、cafe.drinkである。この飲物の例に対する例示的な適用は以下に述べる。CML中のすべてのフィールド名は常に完全に限定されることに留意されたい。すなわち、フィールド名内には、関連するジェスチャが発生するネスティング・レベルに基づく暗黙的な階層はない。

##### 【0092】E. CML属性

CMLインスタンスは、次のXML属性を有することが

できる。別の方法で述べていない限り、すべての属性はオプションである。

- ( i ) node\_id このCMLノードに対する固有の識別子。属性node\_idが要求される。
- ( ii ) title CMLインスタンスにタイトルを指定する、人間が読めるメタデータ・ストリング。
- ( iii ) name CMLインスタンス内でインスタンス化されたすべてのフィールド値に対して名前空間を確立するのに使用される名前。
- ( iv ) action CMLのターゲット・アクションであるURL (Uniform Resource Locator) を指定する。
- ( v ) style 関連するXSLスタイル・シートのURI。指定がない限り、CMLインタープリタは、モダリティ独立のCMLインスタンスをモダリティ特定のコード化に変換するために、デフォルトで一般スタイル・シートに設定する。属性styleは、CML作成者がシステム全体のスタイル規則をオーバーライドまたは特殊化することを可能にする。

【0093】F. CMLコンポーネント  
CMLインスタンスは、「会話型ジェスチャ」を表す。前述のように、会話型ジェスチャは、ダイアログの基本的な構成単位であり、対話論理をモダリティ独立の形でカプセル化する。複雑な会話型コンポーネント (ダイアログ・コンポーネントまたはダイアログ・モジュールとも呼ばれる) は、後続のサブセクションで述べるより基本的な会話型ジェスチャを統合することによって構築される。これらの複雑な会話型コンポーネントは通常、例えば電話番号を得る、住所を得るなどのタスク指向である。基本的な会話型ジェスチャのCML記述は、所望のレベルの複雑性までネストすることができる。ネスティングの他にも、複雑な会話型コンポーネントは、基本的な会話型ジェスチャを並列で、または逐次的に、あるいはその両方で結合することによっても得られる。また、複雑な会話型コンポーネントは、命令型ジェスチャ、例えばCFC (Conversational Foundation Class) を結合することによっても得られ、これは以下に述べる。あらゆるCMLジェスチャがXML要素であるものの、その逆は真ではないことにも留意されたい。すなわち、本仕様中に定義するあらゆるXML要素がCMLジェスチャであるというわけではない。多くのCMLジェスチャは、サブ要素を使用して所与のジェスチャの基礎構造をカプセル化する。後続のセクションでは、「ジェスチャ」であるCML要素は、ジェスチャ・メッセージというタイトルのサブセクション中のようにマークされる。

【0094】CMLは、新たに出現する他のW3C規格、例えばXHTML (Extensible HyperText Markup Language) などと相互動作するように設計される。したがってCML要素は、HTML、MATHMLなどのような他のマークアップ言語から要素を再創作するのではなく、適切なところで再利用する。このような要素は、

CMLインスタンス中に埋め込まれると、例えばhtml:emなどに完全に限定される。以下の第1のサブセクションは、様々なCML構成単位の共通面を紹介し、後続のサブセクションは、各構成単位を詳細に記述する。各CML基本要素が基本的な会話型ジェスチャを取り込み、XML属性がより特殊化された挙動をコード化するのに使用されることに留意されたい。したがって、例えば、イエス／ノーの質問はCML基本要素であり、ユーザの確認を必要とするイエス／ノーの質問は、この基本要素を改良したものである。

【0095】CMLはジェスチャの上書きおよびジェスチャの拡張を可能にするので、CMLの特定の実施形態中に提供されるCMLジェスチャの基本セットがどんなものであるかは問題ではない。本明細書で提供されるセットおよび規則は、どんなレガシー・ページおよび対話の実施も可能にする。

【0096】CMLジェスチャは、以下の共通XML属性を共用する。  
action ジェスチャの完了時に行われるアクション。属性actionは、link、return、またはsubmitのうちの1つとすることができる。

【0097】( i ) ジェスチャMessage  
会話型ジェスチャmessageは、ユーザに情報メッセージを伝達するのに使用される。ジェスチャmessageは通常、表示される文字列または話されるプロンプトとしてレンダリングされる。話されるメッセージの一部は、CMLインタープリタによってホスティングされている様々な個々のインフォウェアの現状態の関数とすることができる (アクセス環境の状態に関するセクション参照)。例: <message node\_id="1">Your <html:em> checking</html:em> account balance is<value name="banking.checking.balance"/>after transferring<value name="banking.checking.transfer"/>to your<value name="banking.creditCard.account"/></message>空の要素valueは、現在の環境からの可変の情報を挿入するのに使用され、アクセス環境の状態に関するセクションで正式に定義する。

【0098】( ii ) ジェスチャHelp  
会話型ジェスチャhelpは、ダイアログにトラブルが起きた場合に表示すべきコンテキスト・ヘルプをカプセル化するのに使用される。ジェスチャhelpは通常、表示されるストリングまたは話されるプロンプトとしてレンダリングされる。メッセージの一部は、CMLインタープリタによってホスティングされている様々な個々のインフォウェアの現状態の関数とすることができる。

例:  
help node\_id"1" You can check your account balances by specifying a particular account. </help>

【0099】( iii ) Final  
CML要素finalは、カプセル化されたジェスチャが首

尾よく完了したときに行われるアクション、例えばユーザ対話に基づいてカプセル化環境を更新することをカプセル化するためにジェスチャ内で使用される。

【0100】(iv) ジェスチャBoolean: イエス/ノーの質問

会話型ジェスチャbooleanは、典型的なイエス/ノーの質問をカプセル化する。ジェスチャbooleanは、messageとして使用すべきプロンプトならびにもしあればデフォルトの応答をカプセル化する。属性require\_confirmation、require\_confirmation\_if\_no、およびrequire\_confirmation\_if\_yes (デフォルトではすべてfalse) によって、インフォウェア・アプリケーションがダイアログを改良することができる。

例:

```
<boolean default="y"
node_id="1"
require_confirm_if_no="true">
<grammar type="text/jsgf">
(yes | yeah) {yes} | (no nay) {no}
</grammar>
<message>
Please confirm that you would like to stay at the
<value href="travelCenter.hotel.selected"/>
</message>
</boolean>
```

【0101】(v) ジェスチャSelect

会話型ジェスチャselectは、ユーザが一組の選択肢から選び取ることを期待される場合のダイアログをカプセル化するのに使用される。これは、プロンプト、デフォルト選択、ならびに合法的な選択肢のセットをカプセル化する。要素selectの属性は、相互排他的選択を達成する (ラジオ・ボタンのグループとして視覚的にレンダリングされる)、範囲から選択する、スクロールバーとして視覚的にレンダリングするなどのためにジェスチャを改良する。selectのサブ要素は、以下のものを含む。

choices

値と参照のいずれかによって埋め込まれた可能な選択肢のリストを含む。要素choicesは、1つまたは複数のchoice要素のリスト、ならびにもしあればデフォルト選択を指定するせいぜい1つのdefault要素を含む。

predicate

選択が満たすべきテストをカプセル化する述語。

help

ダイアログが行き詰まった場合に提供すべきヘルプ。

error

predicateが失敗の場合に使用すべきメッセージを含む。

例:

```
<select name="portfolio.fund"
node_id="1"
```

```
require predicate="yes">
```

```
<message node_id="2">
```

Which of your positions would you like to check?

```
</message>
```

```
<help>
```

You can specify the names of stocks or funds you own and we will report your current position.

```
</help>
```

```
<choices>
```

```
<var name="possibleChoices"/>
```

```
<default value="possibleChoices">
```

Check the position of all holdings</default>

```
</choices>
```

```
<predicate>
```

```
<conduction>
```

fund in possibleChoices

```
</condition>
```

```
<error>
```

Sorry, you do not appear to own any shares in

```
<var name="portfolio.fund"/>
```

```
</error>
```

```
</predicate>
```

```
</select>
```

【0102】(vi) Predicate

要素predicateは、特定の会話型ジェスチャの結果を妥当性検査するための規則をカプセル化するためにCML中で使用される。テスト述語は、その開示を参照により本明細書に組み込むW3Cからのxpath仕様すなわちXML Path Language, W3C Proposed Recommendation中に定義される表現構文および意味論を使用して単純な条件文として表される。http://www.w3.org/tr/xpathを参照されたい。xpathは、文書ツリーの異なる部分にアクセスするための表現構文を指定し、アプリケーション・バックエンドへの呼出しを必要とする妥当性検査は、個別に扱われる。

【0103】predicate要素を含む会話型ジェスチャは、テストが失敗した場合に適切な属性を介して行うべきアクションを限定する。

【0104】(vii) Grammar

CMLサブ要素grammarは、VoiceXML中で要素grammarの後にモデリングされる。サブ要素grammarは、文法フラグメントをコード化し、サブ要素helpが、どの発話が許容されるかをユーザに示すために再生される適切なヘルプ・メッセージをカプセル化する。適切な場合に、CMLジェスチャは、CMLインタープリタによってより複雑な文法に組み立てられる文法フラグメントを提供することができる。

【0105】サブ要素grammarは、入力、特にスピーチを処理する規則として一般化することができる。これらの規則は、厳密にすることができる、または、処理に使



用すべき遠隔リソース (URL) を記述して、どのデータ・ファイルでどの処理を行い、どのアドレスにどんな結果を返さなければならないかを記述する引数を、これらのリソースに渡すために提供することができる。一般に文法は、適正範囲内に定義するかまたはURLを介して定義することができる。

【0106】さらに、この処理を、オブジェクト・タグ、例えば<object>...</object>によって宣言することも可能である。オブジェクト・タグによって、CFC (Conversational Foundation Classes) またはCAP (Conversational Application Platform) サービスをロードすることができる (例えば、上に参照したUS99/22927 (整理番号Y0999-111) として識別されるPCT国際特許出願を参照されたい。ここでCAPはCVMすなわち会話型仮想マシン (Conversational Virtual Machine) に相当する。) 引数は、XML属性および変数を使用してオブジェクトに渡すことができる。結果は、同様の変数プレースホルダを介して返すことができる。これにより、これらのオブジェクト呼出しは、環境にアクセスしてそれを修正することができる。

【0107】オブジェクトは、以下の値をとることのできる属性executeによって限定することができる。すなわち、parallel (並列でブロックされずに実行され、環境への進行中の影響を、その完了前の実行中に知らせることができる)、asynchronous (非同期でブロックされずに実行され、環境の更新が完了したときにイベントを介して知らせる)、blocking (環境の更新および継続の前にブラウザがオブジェクト呼出しの完了を待機する) である。

【0108】処理を分散させるのに必要なすべての情報は、上で参照したUS99/22925 (整理番号Y0999-113) として識別されるPCT国際特許出願に記載されており、これは、会話型アプリケーションの分散を可能にする体系およびプロトコルを定義する。したがって、この国際特許出願は、そのような分散をどのように行うことができ、それがどのようにこの場合のクライアント・ブラウザとサーバ・ブラウザとの間、ならびにローカル・エンジンとサーバ・エンジンとの間に処理を分散させることを可能にするかを記述する。これにより、ネットワークにわたる入力/出力イベントの処理の分散が可能である。

【0109】(viii) ジェスチャMenu

ジェスチャmenuは、ジェスチャselectの特殊なケースである。ジェスチャmenuは、ユーザがあるアプリケーションの異なるサブパートをナビゲートするのを助けるダイアログをカプセル化するのに使用される。ジェスチャselectを使用して、同じ結果を達成することができる。しかし、明示的なmenuジェスチャを有することにより、作成者は、selectジェスチャが使用されている理由に関し

てより意味的な情報を提供することができる。以下の例で、要素menuは、属性actionがlinkに設定されている要素selectに相当することに留意されたい。

例:

```
<menu name="main">
  <choice value="#query">Ask a question</choice>
  <choice value="#browse">Browse available categories</choice>
</menu>
```

各選択における属性valueの値は、その選択に対するURIターゲットを指定する。

【0110】(ix) ジェスチャUser Identification  
会話型ジェスチャuser\_identificationは、ユーザ・ログインおよび認証をカプセル化するのに使用される。これは総称的なものとされ、スタイル規則を介して特定のユーザ対話環境用に特殊化される。

【0111】サブ要素userおよびidentifyは、ユーザ名および認証の情報を得るための会話型ジェスチャをカプセル化する。要素predicateは、ユーザが首尾よく認証されたかどうかを確認するためのテストを提供する。

例:

```
<user_identify name="login"
  require_predicate="yes"
  on_fail="retry"
  node_id="2">
  <message node_id="3">
    To use this service, you first need to login using
    your name and personal identification.
  </message>
  <user name="userid"
    node_id="4">
    what is your user id?
  </user>
  <identify name="pin"
    node_id="4">
    Please provide your user authentication.
  </identify>
  <predicate>
  <conditon>
    backend.authenticate(user id,pin)
  </condition>
  <predicate>
  <error>
    Sorry, login for <var name="userid"/>
    with identification <var name="pin"/> failed.
  </error>
</user_identify>
```

このジェスチャに対する変形は、例えば、識別ジェスチャ (例えばその人が誰かを識別する)、検証ジェスチャ (例えば要求者の認証)、スピーチ生物測定法 (例えば



米国特許第5897616号)の間の明示的な区別に有用となることができる。

【0112】(x) ジェスチャ制約付きのInput  
CMLは、日付や通貨などのユーザ入力を得るための、いくつかの定義済みダイアログ・コンポーネントを提供する。通常、このような入力は、これまでに列挙した様々な選択ジェスチャよりも拡張可能であり、単純な編集フィールドを介して従来の視覚インタフェース中で実現される。しかし、このような入力ジェスチャに対するドメイン特定の制約をカプセル化することは、口頭の対話を構築する際に有利である。また、このようなドメイン特定の制約は、通常、サーバにサブミットする前にユーザ入力の妥当性検査を行うHTMLページ内のクライアント側スクリプトとして、今日のWWWインタフェース中で実施されていることにも留意されたい。CMLでは、今日のWWW上で広く使用されているこれらの入力ジェスチャを、標準的なユーザ・レベルのタスクを行うために形式化する。CMLはまた、この基本的な入力ジェスチャのセットを時の経過に伴って拡張することのできる拡張機構も提供する。このリスト中に定義するすべてのCML要素はジェスチャであることに留意されたい。

- (1) Date 日付を指定する。
- (2) Time 時間を指定する。
- (3) Currency 通貨額を指定する。
- (4) Credit Card クレジット・カード(カード・タイプ、カード番号、有効期限を含む)を指定する。
- (5) Phone 電話番号を指定する。
- (6) Email 電子メール・アドレスを指定する。
- (7) URL URLを指定する。
- (8) Snail Address 郵便番号、国/州/市、通りを含めた「スネイル・メール」アドレスを指定する。

【0113】制約付きの入力ジェスチャは、文法を他の入力フィールドに渡すことによって容易に拡張することができる。このジェスチャはさらに、ローカライズできる(すなわち、国際化でき、地域的な趣を持たせることのできる)トランスコーディング規則に関連付けられることもできることに留意されたい。これは、すべてのジェスチャおよびジェスチャ・ベースのトランスコーディング規則に実際に拡張可能なステートメントである。ロケーション(すなわち電話番号、IPアドレスの起点、ユーザに関して分かっている(ユーザのローカル装置/ブラウザ上にあるかまたはクッキーを介してサーバに送信された)プリファレンス)に基づいて、ジェスチャを別の言語で表すことができ(すなわち、「Select yes or no」が「Selectionnez oui ou non」になるなど)、あるいは地理に適合させることができる(例えば、zipコードが郵便番号になる)。

【0114】(xi) ジェスチャ制約のないInput  
会話型ジェスチャinputは、入力制約がより複雑な(ま

たはおそらく存在しない)場合の、得られたユーザ入力に使用される。このジェスチャは、ユーザ・プロンプト、要求されている情報のアイテムに関するアプリケーション・レベルの意味構造、およびもしかすると入力の妥当性をテストするためのpredicateをカプセル化する。アプリケーション特定の意味制約を伴うジェスチャinputは、前のセクションで考察した、組み込まれた制約付き入力のジェスチャのセットを拡張する手段を提供することに留意されたい。

例:

```
<Input node_id="1">
<Message>...</Message>
</Input>
```

【0115】(xii) ジェスチャSubmit

会話型ジェスチャsubmitは、パッケージされて、含んでいるCMLインスタンスから返されるべき、環境からのコンポーネントを指定する。これはまた、使用すべきプロンプト、ならびにカプセル化された環境の状態をサブミットすべきターゲットURIもカプセル化する。

例:

```
<submit target="uri">
<env name="location.state"/>
<env name="location.city"/>
```

</submit>サブ要素envは、囲んでいるジェスチャがサブミットすべき環境のコンポーネントを指定する。

【0116】様々なCML属性およびコンポーネントを以上に説明したが、CMLのこの実施形態の他の態様を説明しながら、その他の属性およびコンポーネントを以下に呈示および定義することを理解されたい。その他の属性およびコンポーネントは、本発明の教示に従って定義される場合があることを理解されたい。すなわち、本発明は、この詳細な説明で述べる特定の属性およびコンポーネントに限定されるものではない。

【0117】G. 結合イベント

CMLは、アプリケーション作成者が「論理入力イベント」、ならびに、そのような論理イベントと、定義された論理イベントをトリガする実際の「物理入力イベント」との間の関連を定義するための、フレキシブルかつ拡張可能な機構を提供する。CMLジェスチャは、定義された論理イベントが受け取られるとき、CMLジェスチャが扱う準備のできている論理イベントを、CML属性triggerを介して宣言し、そのトリガ・リスト中にマッチング・イベントを有する最も近くの囲んでいるジェスチャが、そのイベントを扱う。CML属性triggerは、ジェスチャがそれに論理的に結合されたイベントによってトリガされるようにする。この機構は、例によって最もよく示される。以下に示すCMLコードの断片では、アプリケーションは、helpを論理入力イベントとして定義し、これを異なる2つのモーダル性にある物理イベントに結合し、最終的に、helpイベントを扱うCML

ジェスチャを宣言する。

例：

```
<cml name="travel">
  <bind-event    logical="help"
                 modality="dtmf"
                 physical="*/>
  <bind-event logical="help"
                 modality="qwerty"
                 physical="h"/>
  <help name="help"
        trigger="help">
    Top-level application help
  </help>
  ...
</cml>
```

CML要素bind-eventは、3つの属性をとる。

- (1) logical 定義されている論理イベントの名前を指定する。
- (2) modality イベントが結合されている対話モデル性を指定する。
- (3) physical 論理イベントに結合させる物理イベントを指定する。

【0118】アプリケーションを構成するCMLジェスチャによって扱われない入力イベントは、helpなどの標準的なプラットフォーム・イベントがデフォルト・ハンドラによって扱われるCMLインタプリタにバブルアップする。バブルアップとは、トリガ値と一致するジェスチャを探索することが、一致するジェスチャがなくなるまで階層的に、最も近くの囲んでいるジェスチャからより高いジェスチャにバブルアップするという意味である。このような場合、トリガは、ブラウザから、さもなければ基礎をなすプラットフォーム（例えばYO999-111の会話型仮想マシン）から提供されるサービスに関連付けるべきである。どれも満たされない場合、そのイベントは無視されるか、または入力理解されず（またはサポートされず）無視されたことを説明するデフォルト・メッセージがユーザに返される。しかしこれらは、ブラウザおよび基礎をなすプラットフォームを実施する選択肢であって、言語の選択肢ではない。機構bind-eventは、プラットフォームの挙動をオーバーライドするように設計され、これは、ユーザ入力をCMLジェスチャにマッピングするための排他的機構として使用されるものではないことに留意されたい。したがって、アプリケーション中の有効なすべての口頭発話を適切なジェスチャに結合するのに要素bind-eventを使用することは反対される。

【0119】さらに、要素bind-event中の属性modalityを省略することにより、指定の物理結合がすべてのモデル性に関連付けられることにも留意されたい。要素bi

nd-event中の属性physicalの値を省略することは、非結合の、すなわち物理イベントに結合されない論理イベントを宣言する。

【0120】H. グループ化ジェスチャおよび定義フォーカス

会話型ジェスチャは、特定のユーザ・インタフェースを実現するために特定のモデル性にされるとき、適切にグループ化されて、ユーザがインタフェースの関係する部分と対話することを可能にする。この断定を理解するために、ユーザ対話をいくつかのHTMLページにわたって分割し、インタフェースの関係する部分が同じページ上に表示されるWWWアプリケーションを考察されたい。同様に、スピーチ・インタフェースは、ユーザがいくつかの関係するコマンドのいずれかを所与の時間に指定することを可能にする。

【0121】こうした形のジェスチャのグループ化は、アプリケーションがオーサリングされているときにもっともよく取り込まれる。このようなグループ化は、モデル性独立の場合もあり、そうでない場合もある。CMLによれば、アプリケーション作成者は、両方の形のグループ化をカプセル化することができる。

【0122】会話型ジェスチャは、CML要素groupを使用してグループ化される。要素groupはさらに、属性id、modality、classによって限定される。属性idは、ジェスチャをグループ化するのに最低限必要である。属性modalityは、もしあれば、指定のグループ化をモデル性特定にすることを宣言する。属性classは、関係する要素をさらに選択してCMLをHTMLのような言語にトランスコーディングするために、HTMLのclass属性と同様の方式で 사용할 ことができる。

【0123】デフォルトで、単一のgroup要素に囲まれたCMLジェスチャは、含まれるいずれかのジェスチャとユーザが対話できるようにするユーザ・インタフェースにマッピングする。HTMLの場合、これは、単一ページにトランスコーディングされているジェスチャとなり、VoiceXMLの場合、これは、並列で活動化されている対応するフォームとなる。

【0124】ジェスチャのグループを並列で活動化することは、混合イニシアチブNLインタフェースを実施する方式であり、所与の時間にサポートされる各コマンド／照会は、ジェスチャの中から構築されたフォーム（すなわちジェスチャのグループはフォームと呼ばれる）を特徴とすることに留意されたい。入力／出力イベントが発生するとき、ブラウザまたは基礎をなすプラットフォームから提供されるダイアログ・マネージャは、活動状態にされた異なるフォーム中のジェスチャが何であるかを推測することになり、それらの関連する属性（ジェスチャに関連付けられた環境変数）を限定できるようにする。フォームのすべての必須属性が値を受け取ると、アクションは一義化されて実行されると見なされる。以下

に述べるが、XFORMSを使用して属性間の追加の制約を表すこともできることに留意されたい。並列活動化に関する考察に対する、上で参照した整理番号Y0998-392によって識別される特許出願と、K.A.Papineni他の「Free-flow dialogmanagement using forms」、Proc. Eurospeech,1999、およびその開示を参照により本明細書に組み込むK.Davies他の「The conversational telephony systemfor financial applications」、Proc. Eurospeech,1999も参照されたい。

【0125】要素groupのインスタンスは、内部グループ要素が、囲んでいる要素中に指定されたものとは異なる値を属性modalityまたはclassに対して指定しない限り、ネストすることができない。

【0126】XFORMS (<http://www.w3.org/MarkUp/Forms/>) のような努力は、同じバックエンドを保持しながら異なる種類のブラウザに向けた表示の置換を容易にする試みにおいて（ただしXFORMSは異なるモーダル性に対処するのに失敗している）、図4に示すようにフォームを3層（プレゼンテーション、論理、データ）に分割することにより、既存のマークアップ言語に関連する問題を解決しようとしてきた。XFORMSデータ層は、アプリケーション開発者がフォームに対してデータ・モデルを定義することを可能にする。開発者は、組込みデータ・モデルを使用することもでき、自分自身のものをロールすることもできる。XFORMSは、XMLスキーマ上で行われている作業の最上部にデータ・モデルを構築している。論理層は、アプリケーション開発者が、例えば現総計に対する、またはあるフィールドがファイルされるのに別のフィールドを必要とする場合の、フィールド間の依存性を定義することを可能にする。XFORMSは、スプレッド・シートおよび既存のフォーム・パッケージへの広く行き渡った親しみの上に構築する軽量表現構文をサポートする。アプリケーション開発者は、追加のフレキシビリティが必要なときに、依然としてスクリプトを呼び出すことができる。プレゼンテーション層は、フォーム制御用のマークアップおよび他のHTMLマークアップからなり、各制御は、データ・モデル中のフィールドに結合される。「getter」および「setter」関数は、正規の表現を内的に保持しながら、プレゼンテーションが例えば日付および通貨に対するユーザのプリファレンスに一致するようにし、したがって、フォーム処理を簡略化する。同じデータ・フィールドが、2つ以上のプレゼンテーション制御を結合されることができる。いずれかの制御で値が変更されると、次いで、他のすべてが自動的に更新される。

【0127】本明細書で説明したように、XFORMSは、プレゼンテーションからデータを分離するためのバックエンド機構を提供する。CMLは、論理およびプレゼンテーションの部分をさらにプレゼンテーション・レンダリング（すなわち、対話情報のないモーダル性独立

のレンダリング）／対話（可能性あるモーダル性依存の表面的入力を足したもの）／コンテンツ（すなわち、バックエンド・データと論理情報を足したものから対話に関係するすべての論理対話を引いたもの）に分離するための機構を提供する。この創意に富んだ概念を図5に示す。前に説明したように、本発明のプログラミング・パラダイムは、プレゼンテーション／モーダル性特定のレンダリングAと、対話Bと、コンテンツおよびバックエンド／アプリケーション論理Cとを分離する。図5はまた、前述のXFORMSのバックエンド機構も示し、データDはバックエンドEから分離している。図6は、どのようにフォーム・ベースの混合イニシアチブNLU

（自然言語理解、natural language understanding）アプリケーションがCMLで書かれるかを表している。文字AおよびCは、図5にあるのと同じアイテムを示す。B'で示したブロックでは、対話／ダイアログ情報がCMLで記述される。この部分は、活動化可能な各トランザクションを実現するために発生する必要のある対話（必須およびオプション）を記述する。このためには、基礎をなす属性データ構造に関連する制約およびデータ・モデルを取り込むXFORMSコンポーネントを追加する。エンジン制御および表面変更の部分は、会話型エンジン、特にダイアログ・マネージャおよびNLUエンジンの挙動を最適化するのに使用される追加の制御パラメータを取り込む。CMLの部分は、前に述べたように他のモーダル性でレンダリングするのに使用できることに留意されたい。ブロックFは、ブロックB'に従って利用できる例示的なフォーム（例えばミューチュアル・ファンド・デモ・フォーム）を示す。

【0128】I. データ・モデルおよびデータ・モデルCMLは、XMLスキーマおよびXMLフォームに関するW3Cワークの結果によって指定されるデータ・モデルまたはデータ・モデル基本要素を定義する。<http://www.w3.org>を参照されたい。

【0129】J. アクセス環境

CMLジェスチャは、集合的に「環境」と呼ばれる変数の集合を定義する。CML文書が横断されるとき、環境中の変数は、成功したユーザ対話から得られる値に束縛される。環境は、以下に述べるが、要素var、value、assignを介してCMLジェスチャ内でアクセスし操作することができる。このような名前はすべて、常に完全に限定されることに留意されたい。

【0130】(i) var 要素varは、現行の環境で、変数を宣言し、（かつ任意選択で）初期設定する（それを初期値に割り当てる）。属性nameは、変数の名前を指定する。初期値は、要素assignに対して指定するのと同じ構文を使用して指定することができる。以下を参照されたい。

【0131】(ii) assign 要素assingは、環境中にすでに存在する変数に値を割り当てる。すなわち、要素as

signは、環境中の値を束縛するのに使用される。属性nameは、束縛すべき変数を指定する。束縛すべき値は、属性exprの値として、xpathによって使用されるのと同じ表現構文を使用して指定することもでき、あるいは、割り当てるべき値は、要素assignのコンテンツとして指定することもできる。要素assignは通常、直接的なユーザ対話によって設定されない中間変数を束縛または更新するのに使用される。

【0132】(iii) value 要素valueは、定義された変数の値を取り出す。すなわち、空の要素valueの属性nameは、その値を環境中で検索すべき変数を指定する。属性nameの値は、部分的にまたは完全に限定された名前であり（名前空間に関する前述のセクション参照）、含んでいるCMLジェスチャのコンテキストで解釈される。

【0133】上に定義したように、変数は、それらを割り当てることができる前に宣言されなければならないことに留意されたい。

#### 【0134】K. CML横断モデル

CMLでオーサリングされたインフォウェアは、複数のユーザ・エージェント間を調停する会話型シェルによってホスティングされ、この会話型シェルを、以後、CMLインタプリタと呼ぶ。横断モデルは、図13および14のコンテキストでさらに考察し、例示することを理解されたい。ユーザ対話は、CMLインタプリタがCMLインスタンスをHTMLやVoiceXMLなどの適切なモーダル性特定の言語にマッピングすることによって進行する。これらのモーダル性特定の表現は、モーダル性特定バージョンのダイアログをレンダリングする適切なユーザ・エージェントに渡される。

【0135】CMLからモーダル性特定の表現への変換は、XSL変換規則（XSLT）に支配されることが好ましい。他の変換機構も使用することに留意されたい。XSLTは、好ましい実施形態に提案される一手段にすぎない。例えば、JSP、Java Server PagesまたはJava Beans、ならびに規則に基づいてジェスチャをそれらのターゲット・レンダリングに変換する他の技術を使用することもできる。このような実施の一例は、各ジェスチャにジャバ・ビーンを関連付けるものである。ジャバ・ビーンは、各モーダル性でそれ自体のレンダリングを（JSPを介して）伝える。したがって、本発明はXSLTに限定されない。いずれの場合にも、これらのXSL規則は、モーダル性特定である。CMLインスタンスを適切なモーダル性特定の表現にマッピングするプロセスで、XSL規則は、モーダル性特定のユーザ対話の実現に必要な情報を追加する。例として、要素selectをVoiceXMLに変換するとき、関係するXSL変換規則は、その会話型ジェスチャに有効な選択肢をカバーする文法の生成を扱う。

【0136】CMLインスタンスをHTMLなどのモー

ダル性特定の表現に変換するプロセスは、単一のCMLノードを出力表現中のノードの集合にマッピングすることになる可能性がある。これらの様々な表現にわたる同期化を助けるために、CML属性node\_idが、所与のCMLノードから得られたすべての出力ノードに適用される。所与のCMLインスタンスが、適切なモーダル性特定のXSL規則によって異なる表現、例えばHTMLおよびVoiceXMLにマッピングされるとき、出力におけるツリーの形は、様々なモーダル性にわたっておそらく非常に異なる。しかし、属性node\_idは、モーダル性特定の各表現から発生源であるCMLノードへの概念上のバックリンクを提供することによって、これらの表現の間で同期をとることを可能にする。上で参照した米国第60/128081号（整理番号Y0999-178）として識別される米国仮特許出願に、マルチモーダル・アプリケーションをしっかりとサポートできるプラットフォーム（マルチモーダル・シェル）をどのように開発するかに関する記述が提供されている。この機構は、次のように動作する。各モーダル性は、それがサポートするコマンドと、それらの実行が他の登録済みモーダル性に与えることになる影響とをマルチモーダル・シェルに登録する。明らかに、ここでの場合、CMLページを解析しジェスチャをトランスコーディングするとき、各ジェスチャは、マルチモーダル・シェル中のデータ構造（すなわちテーブル）中に保持される。所与のモーダル性のI/Oイベント時に、活動化されたジェスチャを見つけるためにnode\_id情報が使用され、テーブル（すなわちCML文書ダイアログ・ツリー）から、活動化されたモーダル性ならびに他のモーダル性への影響が即座に見つかる（すなわち、各ビューを更新するかまたはCMLサーバ上の新しいページを取り出す）。

【0137】ユーザ対話が進行するにつれ、現行のCMLインスタンスによって環境中で定義された変数は、妥当性検査された値に束縛される。まず、登録されたモーダル性特定のユーザ・エージェントの1つでこの束縛が発生する。登録されたユーザ・エージェントは、更新された環境および完了したばかりのジェスチャのnode\_idからなる適切なメッセージを会話型シェルに送る。更新された束縛がCMLインタプリタに伝えられると、CMLインタプリタは、完了したばかりのジェスチャのnode\_idで、登録されたすべてのユーザ・エージェントにメッセージを送る。登録されたユーザ・エージェントは、このメッセージを受け取ると、まずそれらのプレゼンテーションに影響する環境の部分に対してCMLインタプリタに照会することによって、それらのプレゼンテーションを更新する。

【0138】L. 特定ユーザ・インタフェース言語へのCMLの変換

CMLは、XSLで表現される変換規則によって、ユーザ・インタフェース（ui）特定のコード化、例えばH

TMLに変換される。このセクションは、XSL変換についてのいくつかの背景材料で始まり、次いで、本発明によってXSLがどのようにCMLおよびマルチモーダル・ブラウザのコンテキストで使用されるかについての例を呈示する。

#### 【0139】(i) XSL変換の背景情報

W3C XSL変換(xslt)仕様は、James Clark編Proposed Recommendation: XSL Transformations (xslt) Version 1.0, reference: W3C Proposed Recommendation 8-October-1999としてリリースされており、この開示を参照により本明細書に組み込む。上に参照したW3C Proposed Recommendationは、W3C Style活動の一部である。具体的には、xslt仕様は、XML文書を他のXML文書にトランスコーディングするための言語であるxsltの構文および意味論を定義する。xsltは、XMLに対するスタイル・シート言語であるXSLの一部として使用するために設計されている。xslt言語中の変換は、xsltによって定義される要素とxsltによって定義されない要素とを含むことのできるXML Recommendation中の名前空間に準拠した、うまく形成されたXML文書として表現される。xsltで表される変換は、ソース・ツリーを結果ツリーに変換するための規則を記述する。変換は、パターンをテンプレートと関連付けることによって達成される。パターンは、ソース・ツリー中の要素と突

き合わせられる。テンプレートは、結果ツリーの一部を生み出すためにインスタンス化される。結果ツリーは、ソース・ツリーとは別個である。結果ツリーの構造は、ソース・ツリーの構造とは全く異なるものとすることができる。結果ツリーを構成する際、ソース・ツリーからの要素は、フィルタリングしてリオーダーすることができる。xsltで表された変換は、スタイル・シートと呼ばれる。xslt仕様は、XMLとHTMLの両方のフォーマットで入手可能である。

#### 【0140】(ii) XSL変換の例

以下は、CMLコード、XSL変換規則と、それぞれの変換から得られたHTML、WML、およびVoiceXMLコードを示すコード化の例である。

【0141】以下のコードは、CMLと、レガシーMLページ(それぞれHTML、VoiceXML、およびWML)を作成するのに使用されていた異なるジェスチャベースのXSL規則とによって書かれたページの完全な例を示す。各ページは、後続の図で示すような特定のレンダリングに関連する。この例は、異なる情報サービスへのアクセス、すなわちニュース、ビジネス、スポーツ、旅行、天気、芸能を提供するサイトのものである。

#### 【0142】(a) CMLコード

これは、例に関連するソースCMLページを記述する。

```
<!--$Id: cnn.cml,v 1.19 2000/02/01 Exp $-->
<!--Description: CNN Mobile In cml -->
<cml name="cnn"
  node_id="1"
  title="CNN Mobile News">
  <menu name="cnn.command"
    node_id="2" >
    <choices node_id="3" >
      <default value="#cnn.query">Select News Stories</default>
      <choice value="#cnn.exit"
        require_confirmation="true">
        Exit </choice>
      <choice value="#cnn.applicationHelp">Help</choice>
    </choices>
  </menu>
  <cml name="cnn.applicationHelp"
    title="About CNN Mobile"
    node_id="4"
    action="return">
  <message
    node_id="5" >
    This application allows you to select and view CNN news stories
  </message>
</cml>
<cml name="cnn.exit"
  node_id="6"
```

```

        title="Exit CNN Mobile News"
        action="submit">
<message node_id="60">
    Thankyou for using the CNN news service
</message>
</cml>
<group node_id="7"
    groupId="query">
<cml name="cnn.query"
    title="Search CNN Mobile News"
    node_id="8" >
<menu name="cnn.query.topic"
    node_id="11"
    title="Topic Selection">
<choices node_id="12" >
    <choice value="#cnn.query.news"> News </choice>
<choice value="#cnn.query.business"> Business </choice>
<choice value="#cnn.query.sports">
    <grammar> (sport | sports" </grammar>
    Sports
</choice>
<choice value="#cnn.query.travel"> Travel </choice>
<choice value="#cnn.query.weather"> Weather </choice>
<choice value="#cnn.query.show">
    <grammar > show [business] </grammar>
    Show Business
</choice>
</choices>
</menu>
</cml>
<cml name="cnn.query.news"
    title="News Channel"
    node_id="13"
    action="submit">
<select name="cnn.query.part">
    <message node_id="9" >
    Which part of today's news would you like to read?</message>
<choices
    node_id="10" >
    <choice value="h"> Headlines</choice>
    <choice value="1"> first story </choice>
<choice value="2"> second story </choice>
    <choice value="3"> third story </choice>
</choices>
</select>
<select name="cnn.query.interest">
    <message node_id="14" >
    Which news category would you like to read?
</message>
<choices node_id="15" >

```

```

    <choice value="business">
      <grammar type="text/jsgf">
        business {BIZ} </grammar>
      Business
    </choice>
    <choice value="africa">
      Africa</choice>
    <choice value="world"> World </choice>
    <choice value="United states"> United states </choice>
    <choice value="europe"> Europe </choice>
    <choice value="Asia"> Asia</choice>
    <choice value="me"> Middle East</choice>
    <choice value="america"> America </choice>
  </choices>
</select>
</cml>
<cml name="cnn.query.business"
  title="Business Channel"
  action="submit"
  node_id="16" >
  <select name="cnn.query.part">
    <message node_id="9" >
      Which part of today's news would you like to read?</message>
    <choices
      node_id="10" >
      <choice value="h"> Headlines</choice>
      <choice value="1"> first story </choice>
      <choice value="2"> second story </choice>
      <choice value="3"> third story </choice>
    </choices>
  </select>
  <select name="cnn.query.interest">
    <message node_id="17">
      Which business category would you like to read?</message>
    <choices node_id="15">
      <choice value="NEWS"> news </choice>
      <choice value="IN"> indexes </choice>
      <choice value="CU"> exchange rates </choice>
      <choice value="MET"> metals </choice>
    </choices>
  </select>
</cml>
<cml name="cnn.query.weather"
  title="Weather Channel"
  action="submit"
  node_id="19" >
  <select name="cnn.query.part">
    <message node_id="9" >
      Which part of today's news would you like to read?</message>
    <choices

```

```

        node_id="10" >
        <choice value="h"> Headlines</choice>
        <choice value="1"> first story </choice>
        <choice value="2"> second story </choice>
        <choice value="3"> third story </choice>
    </choices>
</select>
<select name="cnn.query.interest">
    <message node_id="20">
        Which region are you interested in?</message>
    <choices node_id="21">
        <choice value="us"> United states </choice>
        <choice value="europe">
            <grammar type="text/jsf"> (euro | Europe) </grammar>
            Europe
        </choice>
        <choice value="JP"> Japan </choice>
        <choice value="AU"> Australia </choice>
        <choice value="AS"> Asia </choice>
    </choices>
</select>
</cml>
<cml name="cnn.query.travel"
    title="Travel Section" action="submit"
    node_id="522" >
    <select name="cnn.query.part">
    <message node_id="9" >
        Which part of today's news would you like to read?</message>
    <choices
        node_id="10" >
        <choice value="h"> Headlines</choice>
        <choice value="1"> first story </choice>
        <choice value="2"> second story </choice>
        <choice value="3"> third story </choice>
    </choices>
    </select>
    <select name="cnn.query.interest">
        <message node_id="23">
            Which city do you want to visit?</message>
        <choices node_id="24">
            <choice value="AMSTERDAM">AMSTERDAM</choice>
            <choice value="COPENHAGEN">COPENHAGEN</choice>
            <choice value="HELSINKI">HELSINKI</choice>
            <choice value="HONGKONG">HONGKONG</choice>
            <choice value="LONDON">LONDON</choice>
            <choice value="OSLO">OSLO</choice>
            <choice value="PRAGUE">PRAGUE</choice>
            <choice value="SINGAPORE">SINGAPORE</choice>
            <choice value="STOCKHOLM">STOCKHOLM</choice>
            <choice value="SYDNEY">SYDNEY</choice>

```



```

        </choices>
    </select>
</cml>
<cml name="cnn.query.sports"
    action="submit"
    title="Sports Channel"
    node_id="25" >
    <select name="cnn.query.part">
    <message node_id="9" >
        Which part of today's news would you like to read?</message>
    <choices
        node_id="10" >
        <choice value="h"> Headlines</choice>
        <choice value="1"> first story </choice>
        <choice value="2"> second story </choice>
        <choice value="3"> third story </choice>
    </choices>
    </select>
    <select name="cnn.query.interest">
    <message node_id="26">
        What sports are you interested in?</message>
    <choices node_id="27">
        <choice value="AS"> Asia </choice>
        <choice value="w"> world </choice>
        <choice value="eu"> europe </choice>
        <choice value="us"> united states </choice>
        <choice value="nba"> NBA </choice>
        <choice value="nfl"> nfl </choice>
        <choice value="EF"> European football </choice>
    </choices>
    </select>
</cml>
<submit target="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi">
    <message node_id="28">
        executing <value name="cnn.command"/>
        for <value name="cnn.query.part"/>
        stories about <value name="cnn.query.interest"/>
        from topic <value name="cnn.query.topic"/>
    </message>
    <env name="cnn.command"/>
    <env name="cnn.query.topic"/>
    <env name="cnn.query.interest"/>
    <env name="cnn.query.part"/>
    </submit>
</group>
<submit target="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi">
</submit>
</cml>

```

【0143】(b) ジェスチャXSL  
以下の例は、CMLページをジェスチャごとにHTML

ページにトランスコーディングするのに使用される、CMLからHTMLへのジェスチャ・ベースのXSL規則

を示す。任意の可能なCMLページをトランスコーディングするのに必要なジェスチャ・ベースのトランスコーディング規則がすべてあるわけではない。これは、この

方法の例示と見なすべきである。XSL構文は、従来のXSLT規則に従う。例えば、<http://www.w3.org/1999/XSL/Transform>を参照されたい。

```

!-$Id: cml2html.xsl,v 1.8 1999/11/12 20:01:11 $-->
<!--Description: Transform CML to HTML -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"

                                xmlns:xt="http://www.jclark.com/xt"
                                version="1.0"
                                extension-element-prefixes="xt">
<xsl:include href="html/cml.xsl"/>
<xsl:include href="html/environment.xsl"/>
<xsl:include href="html/output.xsl"/>
<xsl:include href="html/selections.xsl"/>
<xsl:include href="common/identity.xsl"/>
</xsl:stylesheet>
<!--$Id: cml.xsl,v 1.13 2000/01/31 Exp $-->
<!--Description: Translate CML element to HTML -->
<!-- Handle case of CML element being the top-level element -->
<xsl:stylesheet

                                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html"/>
<xsl:template match="/cml">
<html>
<head>
<META http-equiv="Content-Type" content="text/html;
charset=iso-8859-1"/>
<title><xsl:value-of select="@title"/></title>
</head>
<body>
<h1>
<a name="{@name}">
<xsl:value-of select="@title"/>
</a>
</h1>
<xsl:choose>
<xsl:when test="@action='submit'">
<form>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:attribute name="action">
<xsl:value-of select="submit/@target"/>
</xsl:attribute>
<xsl:apply-templates/>
<p>
<INPUT TYPE="SUBMIT" VALUE="{@name}"/>
</p>
</form>
</xsl:when>

```

```

        <xsl:otherwise>
            <div node_id="{@node_id}"
                name="{@name}" ">
                <xsl:apply-templates/>
            </div>
        </xsl:otherwise>
    </xsl:choose>
</body>
</html>
</xsl:template>

<xsl:template match="cml[@action='submit']">
    <h2> <a name="{@name}" ">
        <xsl:value-of select="@title"/> </a>
    </h2>
    <form>
        <xsl:attribute name="node_id">
            <xsl:value-of select="@node_id"/>
        </xsl:attribute>
        <xsl:attribute name="action">
            <!-- for rea, we should process submit node to
            cons up target uri -->
            <xsl:value-of select="../../submit/@target"/>
        </xsl:attribute>
        <xsl:apply-templates/>
    <p>
        <INPUT TYPE="SUBMIT" VALUE="{@name}" "/>
    </P>
    </form>
</xsl:template>

<xsl:template match="cml">
    <h2 node_id="{@node_id}" ">
    <a name="{@name}" ">
        <xsl:value-of select="@title"/> </a>
    </h2>
    <xsl:apply-templates/>
    <xsl:if test="@action='return'">
        <p>
            <a name="{concat('#', /cml/@name)}">
                Back
            </a>
        </p>
    </xsl:if>
</xsl:template>

<xsl:template match="group">
    <div groupId="{@groupId}"
        modality="{@modality}"
        class="{@class}" ">
        <xsl:apply-templates/>
    </div>

```

```

</xsl:template>
<xsl:template match="submit"/>
</xsl:stylesheet>
<!--$Id: environment.xsl,v 1.2 2000/02/01 Exp $ -->
<!--Description: Process CML environment constructs -->
<xsl:stylesheet
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="final">
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="var">
    <input type="hidden" name="{@name}" value="{@expr}"/>
  </xsl:template>
  <xsl:template match="assign">
    <input name="{@name}" type="hidden">
      <xsl:attribute name="value">
        <xsl:choose>
          <xsl:when test="@expr=''">
            <xsl:value-of select="."/node()"/>
          </xsl:when>
          <xsl:otherwise>
            <xsl:value-of select="@expr"/>
          </xsl:otherwise>
        </xsl:choose>
      </xsl:attribute>
    </input>
  </xsl:template>
  <xsl:template match="value">
    <b><xsl:value-of select="@name"/></b>
  </xsl:template>
</xsl:stylesheet>
<!--$Id: output.xsl,v 1.3 1999/11/12 20:07:23 Exp $-->
<!-- Description: Transformation rules for CML gestures that -->
<!-- primarily output information -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="message">
    <P>
      <xsl:attribute name="node_id">
        <xsl:value-of select="@node_id"/>
      </xsl:attribute>
      <xsl:apply-templates/>
    </P>
  </xsl:template>
  <!-- eventually generate pop-up help via javascript -->
  <xsl:template match="help">
    <P>
      <xsl:attribute name="node_id">
        <xsl:value-of select="@node_id"/>
      </xsl:attribute>
      <xsl:apply-templates/>
    </P>
  </xsl:template>

```

```

    </P>
  </xsl:template>
</xsl:stylesheet>
<!--$Id: selections.xml,v 1.8 2000/01/31 17:50:34 $-->
<!--Descriptions: Transform CML selection gestures to HTML -->
<xsl:stylesheet
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="menu">
    <xsl:if test="@title!=''">
      <h2>
        <a name="#{@name}">
          <xsl:value-of select="@title"/>
        </a>
      </h2>
    </xsl:if>
    <xsl:apply-templates select="message"/>
    <ol node_id="{@node_id}">
      <xsl:for-each select="choices/choice|choices/default">
        <li>
          <a href="{@value}">
            <xsl:apply-templates/>
          </a>
        </li>
      </xsl:for-each>
    </ol>
  </xsl:template>
  <xsl:template match="select">
    <xsl:apply-templates select="message"/>
    <select name="{@name}">
      <xsl:apply-templates select="choices"/>
    </select>
  </P>
</xsl:template>
  <xsl:template match="choices">
    <xsl:apply-templates/>
  </xsl:template>
  <xsl:template match="choice|default">
    <option>
      <xsl:attribute name="value">
        <xsl:value-of select="@value"/>
      </xsl:attribute>
      <xsl:if test="name(.)='default'">
        <xsl:attribute name="checked"/>
      </xsl:if>
      <xsl:apply-templates/>
    </option>
  </xsl:template>
  <xsl:template match="grammar" />
</xsl:stylesheet>
<!--$Id: identity.xml,v 1.1 1999/11/08 18:05:26 Exp $-->

```

```

<!-- Description: Identity transform for use in other sheets-->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="*|@*">
    <xsl:value-of select="."/>
    <xsl:copy>
      <xsl:apply-templates select="@*" />
      <xsl:apply-templates select="node()" />
    </xsl:copy>
  </xsl:template>
</xsl:stylesheet>

```

【 0144 】 ( c ) HTML ソース

HTML ソース・ページを記述する。得られた歓迎 G U

以下は、( CML から HTML への ) X S L ソースを C  
ML ソース・ページ上に適用することによって得られる

I ページを HTML ブラウザで見たものを、図 7 から 9  
に示す。

```

<!DOCTYPE html PUBLIC "-//M13C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<META http-equiv=" Content-Type" content=" text/html; charset=iso-8859-
1 " >
<title>CNN Mobile News</title>
</head>
<body>
<h1>
<a name="cnn">CNN Mobile News</a>
</h1>
<div node_id="1" name="cnn">
<ol node_id="2">
<li>
<a href="#cnn.query">Select News Stories</a>
</li>
<li>
<a href="#cnn.exit">
Exit </a>
</li>
<li>
<a href="#cnn.applicationHelp">Help</a>
</li>
</ol>
<h2 node_id="4">
<a name="cnn.applicationHelp">About CNN Mobile</a>
</h2>
<p node_id=" 5 " >
This application allows you to select and view CNN news stories
</p>
<p>
<a name="#cnn">
Back
</a>
</p>
<h2>
<a name="cnn.exit">Exit CNN Mobile News</a>

```

```

</h2>
<form node_id= " 6 " action="http://raman.almaden.ibm.com/cgi-bin/cnn.cg
i?command=exit" >
    <P node id="60">
        Thankyou for using the CNN news service
    </P>
<p>
<INPUT TYPE="SUBMIT" VALUE="cnn.exit">
</p>
</form>
    <div groupId="query" modality="" class="">
        <h2 node_id="8">
<a name="cnn.query">Search CNN Mobile News</a>
</h2>
        <h2>
<a name="#cnn.query.topic">Topic Selection</a>
</h2>
<ol node_id="11">
<li>
<a href="#cnn.query.news"> News </a>
</li>
<li>
<a href="#cnn.query.business"> Business </a>
</li>
<li>
<a href="#cnn.query.sports">
        Sports
    </a>
</li>
<li>
<a href="#cnn.query.travel"> Travel </a>
</li>
<li>
<a href="#cnn.query.weather"> Weather </a>
</li>
<li>
<a href="#cnn.query.show">
        Show Business
    </a>
</li>
</ol>
        <h2>
<a name="cnn.query.news">News Channel</a>
</h2>
<form node_id=" 13 " action="http://raman.almaden.ibm.com/cgi-bin/cnn.cg
i?command=search" >
    <P node_id="9">
        Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
    <option value="h"> Headlines</option>

```

```

        <option value="1"> first story </option>
        <option value="2"> second story </option>
        <option value="3"> third story </option>
    </select>
<p>
</p>
    <P node_id="14">
        Which news category would you like to read?
    </P>
<select name= " cnn.query.interest" >
    <option value="business">
        Business
    </option>
    <option value="africa">
        Africa</option>
    <option value="world"> World </option>
    <option value="United states"> United states </option>
    <option value="europe"> Europe </option>
    <option value="Asia"> Asia</option>
    <option value="me"> Middle East</option>
    <option value="america"> America </option>
</select>

<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.news">
</p>
</form>
    <h2>
<a name= " cnn.query.business" >Business Channel</a>
</h2>
<form node_id=" 16 " action="http://raman.almaden.ibm.com/cgi-bin/cnn.cg
i?command=search" >
<P node_id="9">
    Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
    <option value="h"> Headlines</option>
    <option value=" 1 " > first story </option>
    <option value="2"> second story </option>
    <option value="3"> third story </option>
</select>

<p>
</p>
    <P node_id="17">
        Which business category would you like to read?</P>
<select name="cnn.query.interest" >
    <option value="NEWS"> news </option>
    <option value="IN"> indexes </option>
    <option value="CU"> exchange rates </option>
    <option value="MET"> metals </option>

```



```

        </select>
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE=" cnn.query.business" >
</p>
</form>
    <h2>
<a name="cnn.query.weather">Weather Channel</a>
</h2>
<form node_id= " 19 " action="http://raman.almaden.ibm.com/cgi-bin/cnn.c
gi?command=search" >
    <P node id="9">
        Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
    <option value="h"> Headlines</option>
    <option value= " 1 " > first story </option>
    <option value="2"> second story </option>
    <option value="3"> third story </option>
</select>
<p>
</p>
    <P node_id="20">
        Which region are you interested in?</P>
<select name= "cnn.query.interest" >
    <option value="us"> United states </option>
    <option value="europe">
        Europe
    </option>
    <option value="JP"> Japan </option>
    <option value="AU"> Australia </option>
    <option value="AS"> Asia </option>
</select>
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.weather">
</p>
</form>
    <h2>
<a name="cnn.query.travel">Travel Section</a>
</h2>
<form node_id=" 22 " action="http://raman.almaden.ibm.com/cgi-bin/cnn.c
gi?command=search" >
    <P node_id="9">
        Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
    <option value="h"> Headlines</option>
    <option value=" 1 " > first story </option>
    <option value="2"> second story </option>

```

```

        <option value="3"> third story </option>
    </select>
<p>
</p>
<P node_id= " 23 " >
    Which city do you want to visit?</P>
<select name="cnn.query.interest">
    <option value="AMSTERDAM">AMSTERDAM</option>
    <option value= " COPENHAGEN " >COPENHAGEN</option>
    < option value = " HELSINKI " >HELSINKI</option>
    <option value="HONGKONG">HONGKONG</option>
    <option value="LONDON" >LONDON</option>
    <option value= " OSLO " >OSLO</option>
    <option value="PRAGUE">PRAGUE</option>
    <option value= " SINGAPORE " >SINGAPORE</option>
    <option value="STOCKHOLM" >STOCKHOLM</option>
    <option value="SYDNEY">SYDNEY</option>
</select>
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.travel">
</p>
</form>
    <h2>
<a name="cnn.query.sports">Sports Channel</a>
</h2>
<form node_id= " 25 " action= "http://raman.almaden.ibm.com/cgi-bin/cnn.
cgi?command=search" >
    <P node_id="9">
        Which part of today's news would you like to read?</P>
    <select name="cnn.query.part">
        <option value="h"> Headlines</option>
        <option value= " 1 " > first story </option>
        <option value="2"> second story </option>
        <option value="3"> third story </option>
    </select>
<p>
</p>
    <P node_id="26">
        What sports are you interested in?</P>
    <select name="cnn.query.interest">
        <option value="AS"> Asia </option>
        <option value="w" > world </option>
        <option value="eu"> europe </option>
        <option value="us"> united states </option>
        <option value="nba"> NBA </option>
        <option value="nhl"> nhl </option>
        <option value="EF"> European football </option>
    </select>

```

```

<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.sports">
</p>
</form>
</div>
</div>
</body>
</html>

```

#### 【 0 1 4 5 】 ( d ) ジェスチャ X S L

以下の例は、CMLページをジェスチャごとにWMLページにトランスコーディングするのに使用される、CMLからWMLへのジェスチャ・ベースのXSL規則を示す。任意の可能なCMLページをトランスコーディング

するのに必要なジェスチャ・ベースのトランスコーディング規則がすべてあるわけではない。これは、この方法の例示と見なすべきである。XSL構文は、従来のXSLT規則に従う。例えば、<http://www.w3.org/1999/XSL/Transform>を参照されたい。

```

<!-- $Id: cml2html.xsl,v 1.9 2000/02/05 19:32:40 Exp $ -->
<!-- Description: Transform CML to HTML -->
- <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xt="http://www.jclark.com/xt" version="1.0" extension-element-
prefixes="xt">
<xsl:include href="html/cml.xsl" />
<xsl:include href="html/environment.xsl" />
<xsl:include href="html/modality.xsl" />
<xsl:include href="html/output.xsl" />
<xsl:include href="html/selections.xsl" />
<xsl:include href="common/identity.xsl" />
</xsl:stylesheet>
<!--$Id: cml.xsl,v 1.13 2000/01/31 Exp $-->
<!--Description: Translate CML element to HTML -->
<!-- Handle case of CML element being the top-level element -->
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html"/>
<xsl:template match="/cml">
<html>
<head>
<META http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/
>
<title><xsl:value-of select="@title"/></title>
</head>
<body>
<h1>
<a name="{@name}">
<xsl:value-of select="@title"/>
</a>
</h1>
<xsl:choose>
<xsl:when test="@action='submit'">
<form>
<xsl:attribute name="node_id">

```

```

<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:attribute name="action">
<xsl:value-of select="submit/@target"/>
</xsl:attribute>
<xsl:apply-templates/>
<p>
<INPUT TYPE="SUBMIT" VALUE="@name"/>
</p>
</form>
</xsl:when>
<xsl:otherwise>
<div node_id="{@node_id}"
name="{@name}">
<xsl:apply-templates/>
</div>
</xsl:otherwise>
</xsl:choose>
</body>
</html>
</xsl:template>
<xsl:template match="cml[@action='submit']">
<h2> <a name="{@name}">
<xsl:value-of select="@title"/> </a>
</h2>
<form>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:attribute name="action">
<!-- for rea, we should process submit node to
cons up target uri -->
<xsl:value-of select="../submit/@target"/>
</xsl:attribute>
<xsl:apply-templates/>
<p>
<INPUT TYPE="SUBMIT" VALUE="{@name}"/>
</p>
</form>
</xsl:template>
<xsl:template match="cml">
<h2 node_id="{@node_id}">
<a name="{@name}">
<xsl:value-of select="@title"/> </a>
</h2>
<xsl:apply-templates/>
<xsl:if test="@action='return'">
<p>
<a name="{concat('#', /cml/@name)}">
Back

```

```

</a>
</p>
</xsl:if>
</xsl:template>
<xsl:template match="group">
<div groupId="@groupId"
modality="@modality"
class="@class">
<xsl:apply-templates/>
</div>
</xsl:template>
<xsl:template match="submit"/>
</xsl:stylesheet>
<!--$Id: environment.xml,v 1.2 2000/02/01 Exp $ -->
<!--Description: Process CML environment constructs -->
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="final">
<xsl:apply-templates/>
</xsl:template>
<xsl:template match="var">
<input type="hidden" name="@name" value="@expr"/>
</xsl:template>
<xsl:template match="assign">
<input name="@name" type="hidden">
<xsl:attribute name="value">
<xsl:choose>
<xsl:when test="@expr=''">
<xsl:value-of select="./node()"/>
</xsl:when>
<xsl:otherwise>
<xsl:value-of select="@expr"/>
</xsl:otherwise>
</xsl:choose>
</xsl:attribute>
</input>
</xsl:template>
<xsl:template match="value">
<b><xsl:value-of select="@name"/></b>
</xsl:template>
</xsl:stylesheet>
<!-- $Id: modality.xml,v 1.1 2000/02/05 19:32:00 Exp $ -->
<!-- Description: Process CML modality constructs -->
- <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
- <xsl:template match="modality[@class='visual']">
<xsl:apply-templates />
</xsl:template>
- <xsl:template match="var">
<input type="hidden" name="@name" value="@expr" />
</xsl:template>

```

```

- <xsl:template match="assign">
- <input name="{@name}" type="hidden">
- <xsl:attribute name="value">
- <xsl:choose>
- <xsl:when test="@expr=''">
<xsl:value-of select="./node()" />
</xsl:when>
- <xsl:otherwise>
<xsl:value-of select="@expr" />
</xsl:otherwise>
</xsl:choose>
</xsl:attribute>
</input>
</xsl:template>
- <xsl:template match="value">
- <b>
<xsl:value-of select="@name" />
</b>
</xsl:template>
</xsl:stylesheet>
<!--$Id: output.xml,v 1.3 1999/11/12 20:07:23 Exp $-->
<!-- Description: Transformation rules for CML gestures that -->
<!-- primarily output information -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="message">
<P>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:apply-templates/>
</P>
</xsl:template>
<!-- eventually generate pop-up help via javascript -->
<xsl:template match="help">
<P>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:apply-templates/>
</P>
</xsl:template>
</xsl:stylesheet>
<!--$Id: selections.xml,v 1.8 2000/01/31 17:50:34 Exp $-->
<!--Descriptions: Transform CML selection gestures to HTML -->
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="menu">
<xsl:if test="@title!=''">
<h2>
<a name="#{@name}">

```

```

<xsl:value-of select="@title"/>
</a>
</h2>
</xsl:if>
<xsl:apply-templates select="message"/>
<ol node_id="{@node_id}">
<xsl:for-each select="choices/choice|choices/default">
<li>
<a href="{@value}">
<xsl:apply-templates/>
</a>
</li>
</xsl:for-each>
</ol>
</xsl:template>
<xsl:template match="select">
<xsl:apply-templates select="message"/>
<select name="{@name}">
<xsl:apply-templates select="choices"/>
</select>
<p/>
</xsl:template>
<xsl:template match="choices">
<xsl:apply-templates/>
</xsl:template>
<xsl:template match="choice|default">
<option>
<xsl:attribute name="value">
<xsl:value-of select="@value"/>
</xsl:attribute>
<xsl:if test="name(.)='default'">
<xsl:attribute name="checked"/>
</xsl:if>
<xsl:apply-templates/>
</option>
</xsl:template>
<xsl:template match="grammar" />
</xsl:stylesheet>

<!--$Id: identity.xsl,v 1.1 1999/11/08 18:05:26 Exp $-->
<!-- Description: Identity transform for use in other sheets-->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="*|@*">
<xsl:value-of select="."/>
<xsl:copy>
<xsl:apply-templates select="@*"/>
<xsl:apply-templates select="node()"/>
</xsl:copy>
</xsl:template>
</xsl:stylesheet>

```

## 【 0 1 4 6 】 ( e ) WMLソース

MLソース・ページを記述する。得られた歓迎GUIページをWMLブラウザで見たものを、図10に示す。

以下は、(CMLからWMLへの)XSLソースをCMLソース・ページ上に適用することによって得られるW

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN" "http://www.wapforum
.org/DTD/wml_1.1.
<wml>
<template>
<do type="prev" label="Back">
<prev/>
</do>
</template>
<card id="cnn.command" title="cnn.command">
<p>
<select name="cnn.command">
<option onpick="#cnn.query">Select News Stories</option>
<option onpick="#cnn.exit">
Exit </option>
<option onpick="#cnn.applicationHelp">Help</option>
</select>
</p>
</card>
<card id="cnn.applicationHelp" title="cnn.applicationHelp">
<p>
This application allows you to select and view CNN news stories
</p>
</card>
<card id="cnn.exit" title="cnn.exit">
<p>
Thankyou for using the CNN news service
</p>
<p align="center">
<a href="cnn.wmls#submit()" />
</p>
</card>
<card id="cnn.query" title="cnn.query">
<p>
<select name="cnn.query">
<option onpick="#cnn.query.news"> News </option>
<option onpick="#cnn.query.business"> Business </option>
<option onpick="#cnn.query.sports">
Sports
</option>
<option onpick="#cnn.query.travel"> Travel </option>
<option onpick="#cnn.query.weather"> Weather </option>
<option onpick="#cnn.query.show">
Show Business
</option>
</select>
</p>
```



```

</card>
<card id="cnn.query.news" title="cnn.query.news">
<p>
Which part of today's news would you like to read?<select name="cnn.query.part">
<option value="h" onpick="cnn.wmls#submit()"> Headlines</option>
<option value="1" onpick="cnn.wmls#submit()"> first story </option>
<option value="2" onpick="cnn.wmls#submit()"> second story </option>
<option value="3" onpick="cnn.wmls#submit()"> third story </option>
</select>
</p>
<p>
Which news category would you like to read?
<select name="cnn.query.interest">
<option value="business" onpick="cnn.wmls#submit()">
Business
</option>
<option value="africa" onpick="cnn.wmls#submit()">
Africa</option>
<option value="world" onpick="cnn.wmls#submit()"> World </option>
<option value="United states" onpick="cnn.wmls#submit()"> United states
</option>
<option value="europe" onpick="cnn.wmls#submit()"> Europe </option>
<option value="Asia" onpick="cnn.wmls#submit()"> Asia</option>
<option value="me" onpick="cnn.wmls#submit()"> Middle East</option>
<option value="america" onpick="cnn.wmls#submit()"> America </option>
</select>
</p>
<p align="center">
<a href="cnn.wmls#submit()" />
</p>
</card>
<card id="cnn.query.business" title="cnn.query.business">
<p>
Which part of today's news would you like to read?<select name="cnn.query.part">
<option value="h" onpick="cnn.wmls#submit()"> Headlines</option>
<option value="1" onpick="cnn.wmls#submit()"> first story </option>
<option value="2" onpick="cnn.wmls#submit()"> second story </option>
<option value="3" onpick="cnn.wmls#submit()"> third story </option>
</select>
</p>
<p>
Which business category would you like to read?<select name="cnn.query.interest">
<option value="NEWS" onpick="cnn.wmls#submit()"> news </option>
<option value="IN" onpick="cnn.wmls#submit()"> indexes </option>
<option value="CU" onpick="cnn.wmls#submit()"> exchange rates </option>
<option value="MET" onpick="cnn.wmls#submit()"> metals </option>
</select>

```

```

</p>
<p align="center">
<a href="cnn.wmls#submit()" />
</p>
</card>
<card id="cnn.query.weather" title="cnn.query.weather">
<p>
Which part of today's news would you like to read?<select name="cnn.query.part">
<option value="h" onpick="cnn.wmls#submit()"> Headlines</option>
<option value="1" onpick="cnn.wmls#submit()"> first story </option>
<option value="2" onpick="cnn.wmls#submit()"> second story </option>
<option value="3" onpick="cnn.wmls#submit()"> third story </option>
</select>
</p>
<p>
Which region are you interested in?<select name="cnn.query.interest">
<option value="us" onpick="cnn.wmls#submit()"> United states </option>
<option value="europe" onpick="cnn.wmls#submit()">
Europe
</option>
<option value="JP" onpick="cnn.wmls#submit()"> Japan </option>
<option value="AU" onpick="cnn.wmls#submit()"> Australia </option>
<option value="AS" onpick="cnn.wmls#submit()"> Asia </option>
</select>
</p>
<p align="center">
<a href="cnn.wmls#submit()" />
</p>
</card>
<card id="cnn.query.travel" title="cnn.query.travel">
<p>
Which part of today's news would you like to read?<select name="cnn.query.part">
<option value="h" onpick="cnn.wmls#submit()"> Headlines</option>
<option value="1" onpick="cnn.wmls#submit()"> first story </option>
<option value="2" onpick="cnn.wmls#submit()"> second story </option>
<option value="3" onpick="cnn.wmls#submit()"> third story </option>
</select>
</p>
<p>
Which city do you want to visit?<select name="cnn.query.interest">
<option value="AMSTERDAM" onpick="cnn.wmls#submit()">AMSTERDAM</option>
<option value="COPENHAGEN" onpick="cnn.wmls#submit()">COPENHAGEN</option>
<option value="HELSINKI" onpick="cnn.wmls#submit()">HELSINKI</option>
<option value="HONGKONG" onpick="cnn.wmls#submit()">HONGKONG</option>
<option value="LONDON" onpick="cnn.wmls#submit()">LONDON</option>
<option value="OSLO" onpick="cnn.wmls#submit()">OSLO</option>
<option value="PRAGUE" onpick="cnn.wmls#submit()">PRAGUE</option>

```

```

<option value="SINGAPORE" onpick="cnn.wmls#submit()">SINGAPORE</option>
<option value="STOCKHOLM" onpick="cnn.wmls#submit()">STOCKHOLM</option>
<option value="SYDNEY" onpick="cnn.wmls#submit()">SYDNEY</option>
</select>
</p>
<p align="center">
<a href="cnn.wmls#submit()" />
</p>
</card>
<card id="cnn.query.sports" title="cnn.query.sports">
<p>
Which part of today's news would you like to read?<select name="cnn.query.part">
<option value="h" onpick="cnn.wmls#submit()"> Headlines</option>
<option value="1" onpick="cnn.wmls#submit()"> first story </option>
<option value="2" onpick="cnn.wmls#submit()"> second story </option>
<option value="3" onpick="cnn.wmls#submit()"> third story </option>
</select>
</p>
<p>
What sports are you interested in?<select name="cnn.query.interest">
<option value="AS" onpick="cnn.wmls#submit()"> Asia </option>
<option value="w" onpick="cnn.wmls#submit()"> world </option>
<option value="eu" onpick="cnn.wmls#submit()"> europe </option>
<option value="us" onpick="cnn.wmls#submit()"> united states </option>
<option value="nba" onpick="cnn.wmls#submit()"> NBA </option>
<option value="nhl" onpick="cnn.wmls#submit()"> nhl </option>
<option value="EF" onpick="cnn.wmls#submit()"> European football </option>
</select>
</p>
<p align="center">
<a href="cnn.wmls#submit()" />
</p>
</card>
</wml>

```

#### 【0147】(f) ジェスチャXSL

以下の例は、CMLページをジェスチャごとにVoiceXMLページにトランスコーディングするのに使用される、CMLからVoiceXMLへのジェスチャ・ベースのXSL規則を例示する。任意の可能なCMLページをト

ランスコーディングするのに必要なジェスチャ・ベースのトランスコーディング規則がすべてあるわけではない。これは、この方法の例示と見なすべきである。XSL構文は、従来のXSLT規則に従う。例えば、<http://www.w3.org/1999/XSL/Transform>を参照されたい。

```

<!-- cml2wml.xml -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<!--
<xsl:output method="html" indent="yes"/>
-->
<xsl:output method="xml" indent="yes" media-type="text/xml"/>
<xsl:template match="/cml">
<xsl:text disable-output-escaping="yes">

```

```

&lt;!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN" "http://www.wapfo
rum.org/DTD/wml_1
</xsl:text>
<wml>
<template>
<do type="prev" label="Back">
<prev/>
</do>
</template>
<xsl:apply-templates/>
</wml>
</xsl:template>
<xsl:template match="cml">
<xsl:choose>
<xsl:when test="menu">
<!-- to avoid <card><card>..</card></card> -->
<card>
<xsl:attribute name="id">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:attribute name="title">
<xsl:value-of select="@name"/>
</xsl:attribute>
<p><select>
<xsl:attribute name="name">
<xsl:value-of select="menu/@name"/>
</xsl:attribute>
<xsl:apply-templates select="menu/message"/>
<xsl:for-each select="menu/choices/choice | menu/choices/default">
<option>
<xsl:attribute name="value">
<xsl:value-of select="@value"/>
</xsl:attribute>
<xsl:attribute name="onpick">#<xsl:value-of select="@value"/></xsl:attri
bute>
<xsl:call-template name="lex"/></option>
</xsl:for-each>
</select>
</p>
</card>
</xsl:when>
<xsl:otherwise>
<card>
<xsl:attribute name="id">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:attribute name="title">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:apply-templates/>

```

```

</card>
</xsl:otherwise>
</xsl:choose>
</xsl:template>
<xsl:template match='cml[@action="submit"]'>
<card>
<xsl:attribute name="id">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:attribute name="title">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:apply-templates/>
<p align="center">
<a>
<xsl:attribute name="href">
<xsl:value-of select="/cml/@name"/>.wmls#submit()</xsl:attribute>
</a>
</p>
</card>
</xsl:template>
<xsl:template match='select'>
<p>
<xsl:apply-templates select="message"/>
<select>
<xsl:attribute name="name">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:for-each select="choices/choice | choices/default">
<option>
<xsl:attribute name="value">
<xsl:value-of select="@value"/>
</xsl:attribute>
<xsl:attribute name="onpick">
<xsl:value-of select="/cml/@name"/>.wmls#submit()</xsl:attribute>
<xsl:call-template name="lex"/></option>
</xsl:for-each>
</select>
</p>
</xsl:template>
<xsl:template match="menu">
<card>
<xsl:attribute name="id">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:attribute name="title">
<xsl:value-of select="@name"/>
</xsl:attribute>
<p>
<select>

```

```

<xsl:attribute name="name">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:apply-templates select="message"/>
<xsl:for-each select="choices/choice | choices/default">
<option>
<xsl:attribute name="value">
<xsl:value-of select="@value"/>
</xsl:attribute>
<xsl:attribute name="onpick">#<xsl:value-of select="@value"/></xsl:attri
bute>
<xsl:call-template name="lex"/></option>
</xsl:for-each>
</select>
</p>
</card>
</xsl:template>
<xsl:template name="lex">
<xsl:for-each select="node()">
<xsl:if test="position()=last()">
<xsl:value-of select="current()"/>
</xsl:if>
</xsl:for-each>
</xsl:template>
<!-- explicitly remove segment -->
<xsl:template match="submit"/>
<xsl:template match="message"/>
</xsl:stylesheet>

```

【 0 1 4 8 】 ( g ) VoiceXMLを作成するためのXS  
Lソース

以下に、VoiceXMLソース・ページを作成するのに使  
用されるXSLソース・コードを記述する。

```

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Tran
sform">
<xsl:output method="html"/>
<xsl:template match="/cml">
<vxml>
<xsl:apply-templates/>
</vxml>
</xsl:template>
<xsl:template match="menu">
<menu>
<xsl:apply-templates select="message"/>
<xsl:attribute name="id">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:apply-templates select="message"/>
<prompt> Say one of <enumerate/> </prompt>
<xsl:for-each select="choices/choice|choices/default">

```

```

<choice>
  <xsl:attribute name="next">#<xsl:value-of select="@value"/></xsl:attribu
te>
  <xsl:apply-templates/>
</choice>
</xsl:for-each>
</menu>
</xsl:template>
<xsl:template match="cml[@action='return']">
  <form>
    <xsl:attribute name="id">
      <xsl:value-of select="@name"/>
    </xsl:attribute>
    <xsl:attribute name="node_id">
      <xsl:value-of select="@node_id"/>
    </xsl:attribute>
    <xsl:apply-templates/>
    <block><goto>
      <xsl:attribute name="next">#<xsl:value-of select="/cml/menu/@name"/></xs
l:attribut
    </goto></block>
    </form>
  </xsl:template>
  <xsl:template match="cml[@action='submit']">
    <form>
      <xsl:attribute name="id">
        <xsl:value-of select="@name"/>
      </xsl:attribute>
      <xsl:attribute name="node_id">
        <xsl:value-of select="@node_id"/>
      </xsl:attribute>
      <xsl:apply-templates/>
    <block>
      <goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi">
        <xsl:if test="select[@name]">
          <xsl:for-each select="select">
            <xsl:attribute name="submit">
              <xsl:value-of select="@name"/>
            </xsl:attribute>
          </xsl:for-each>
        </xsl:if>
      </goto>
    </block>
    </form>
  </xsl:template>
  <xsl:template match="select">
    <field>
      <xsl:attribute name="name">
        <xsl:value-of select="@name"/>
      </xsl:attribute>

```

```

<xsl:attribute name="node_id">
<xsl:value-of select="../@node_id"/>
</xsl:attribute>
<xsl:if test="message">
<prompt>
<xsl:value-of select="message"/>
Say one of <enumerate/>
</prompt>
</xsl:if>
<grammar>
<xsl:for-each select="choices/choice|choices/default">
<xsl:call-template name="lex"/>
<xsl:if test="following-sibling::choice">|</xsl:if>
</xsl:for-each>
</grammar>
</field>
</xsl:template>
<xsl:template match="message">
<field><prompt>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:apply-templates/>
</prompt>
</field>
</xsl:template>
<xsl:template match="help">
<help>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:apply-templates/>
</help>
</xsl:template>
<xsl:template match="grammar"/>
<xsl:template match="submit"/>
<xsl:template name="lex">
<xsl:for-each select="node()">
<xsl:if test="position()=last()">
<xsl:value-of select="current()"/>
</xsl:if>
</xsl:for-each>
</xsl:template>
</xsl:stylesheet>

```

#### 【0149】(h) VoiceXMLソース

以下に、(CMLからVoiceXMLへの)XSLソースをCMLソース・ページ上に適用することによって得られるVoiceXMLソース・ページを記述する。得られる

歓迎スピーチ・ダイアログは、VoiceXMLブラウザで呈示されるとき、異なるオプション間で音声によって選択するためのダイアログを最初にユーザに呈示する。

```

<vxml>
<menu id="cnn_command" node_id="2">

```



```

<prompt> Say one of <enumerate></enumerate></prompt><choice next="#cnn_q
uery">Select News
Exit </choice><choice next="#cnn_applicationHelp">Help</choice>
</menu>
<form id="cnn_applicationHelp" node_id="4">
<field><prompt node_id="5">
This application allows you to select and view CNN news stories
</prompt></field>
<block><goto next="#cnn"></goto></block>
</form>
<form id="cnn_exit" node_id="6">
<field><prompt node_id="60">
Thankyou for using the CNN news service
</prompt></field>
<block><goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi"></goto>
</block>
</form>
<menu id="cnn_query" node_id="11">
<prompt> Say one of <enumerate></enumerate></prompt><choice next="#cnn_q
uery_news"> News <
Sports
</choice><choice next="#cnn_query_travel"> Travel </choice><choice next=
"#cnn_qu
Show Business
</choice>
</menu>
<form id="cnn_query_news" node_id="13">
<field name="cnn_query_part" node_id="13"><prompt>
Which part of today's news would you like to read?</prompt><grammar> Hea
dlines!
<field name="cnn_query_interest" node_id="13"><prompt>
Which news category would you like to read?
</prompt><grammar>
Business
|
Africa! World ! United states ! Europe ! Asia! Middle East! America </gr
amma
<block><goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi" submit=
"cnn_query_int
</form>
<form id="cnn_query_business" node_id="16">
<field name="cnn_query_part" node_id="16"><prompt>
Which part of today's news would you like to read?</prompt><grammar> Hea
dlines!
<field name="cnn_query_interest" node_id="16"><prompt>
Which business category would you like to read?</prompt><grammar> news !
indexes
<block><goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi" submit=
"cnn_query_int
</form>

```

```

<form id="cnn_query_weather" node_id="19">
<field name="cnn_query_part" node_id="19"><prompt>
Which part of today's news would you like to read?</prompt><grammar> Hea
dlines!
<field name="cnn_query_interest" node_id="19"><prompt>
Which region are you interested in?</prompt><grammar> United states |
Europe
| Japan | Australia | Asia </grammar></field>
<block><goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi" submit=
"cnn_query_int
</form>
<form id="cnn_query_travel" node_id="22">
<field name="cnn_query_part" node_id="22"><prompt>
Which part of today's news would you like to read?</prompt><grammar> Hea
dlines!
<field name="cnn_query_interest" node_id="22"><prompt>
Which city do you want to visit?</prompt><grammar>AMSTERDAM|COPENHAGEN|H
ELSINKI!
<block><goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi" submit=
"cnn_query_int
</form>
<form id="cnn_query_sports" node_id="25">
<field name="cnn_query_part" node_id="25"><prompt>
Which part of today's news would you like to read?</prompt><grammar> Hea
dlines!
<field name="cnn_query_interest" node_id="25"><prompt>
What sports are you interested in?</prompt><grammar> Asia | world | euro
pe | uni
<block><goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi" submit=
"cnn_query_int
</form>
</vxml>

```

【0150】(iii) 密なマルチモーダル・ブラウジングおよびマルチデバイス・ブラウジング上に述べ、かつ、以下により詳細に述べるように、異なるモーダル性を密に同期させることができる。例えば、同じ装置上の音声およびGUIとすることもでき、HTMLまたはWMLブラウザ上のGUIと同期をとられた電話上の音声とすることもできるなどである。

#### 【0151】M. 表面変更

モーダル性特定の表面的コンテンツまたはパラメータを、モーダル性特定のXML構文を使用して追加することができる。モーダル性特定のジェスチャは、モーダル性修飾詞を有するモーダル性特定のXML構文を使用して追加することができる。その他のモーダル性は、これらのコンポーネントを無視するか、または他のもの（例えばキャプション）で置換することができる。

#### 【0152】(i) モーダル性特定の情報

CMLは、対話論理および会話型アプリケーション・フローを指定するための宣言的かつモーダル性独立のマー

クアップ言語に設計される。しかし、アプリケーション作成者がそのうちに、カスタム・プレゼンテーションを達成するためにCMLアプリケーションにモーダル性特定のコンテンツを追加したいと思うであろうことは理解される。CMLは、特定のモーダル性用とされるマークアップの断片をカプセル化するのに使用される要素modalityによって、これを可能にする。このようなモーダル性特定の断片は、指定されたモーダル性にしか表れないことになることに留意されたい。したがって、作成者は、このようなモーダル性特定の断片を、絶対に必要と思われる場合、さらには作成者が他のモーダル性用の代替断片を提供するかまたは他のどんなモーダル性も気にかけない場合だけに使用することが推奨される。XML属性のclassおよびmoduleによって限定される要素modalityは、以下のように定義される。

Class この断片が適用されるモーダル性のクラスを指定する。

module この断片を受け入れることのできるマークアップ

ブ言語モジュールを指定する。

なるHTML特定の断片である。

【 0 1 5 3 】 以下は、視覚表現までに通過されることに

```
<modality class="visual" module="html-basic">
<LINK REL="stylesheet"
  HREF="cnn.css"
  TYPE="text/css"/>
</modality>

  The following is an example of a cosmetized CML page:
<!--$Id: cnn.cml,v 1.21 2000/02/05 20:08:27 Exp $-->
<!--Description: CNN Mobile In cml -->
<cml name="cnn"
  node_id="1"
  title="CNN Mobile News">
<modality class="visual" module="html-basic">
  <LINK REL="stylesheet"
    HREF="cnn.css"
    TYPE="text/css"/>
</modality>
<modality class="visual" module="html">
<TABLE BORDER="0" WIDTH="600" CELSPACING="0" CELLPADDING="0"><TR>
  <TD WIDTH="122" VALIGN="TOP"><a H
    <IMG SRC="http://cnn.com/images/1999/10/cnnstore.gif"
      WIDTH="120" HEIGHT="60" BORDER="1" AL
  <TD WIDTH="8" VALIGN="TOP"><a HREF="http://cnn.com/ads/
    e.market/">
    <IMG SRC="http://cnn.com/images/1998/05/homepage/ad.
      info.gif" WIDTH="7" HEIGHT="62" BORDER=
  <TD WIDTH="470" VALIGN="TOP">
    <a
      HREF= " http: / / cnn. com/event.ng/Type=click%26RunID=
        11875%26ProfileID=34%2 6AdID=13042%2 6Group:
      target="_top">
        
        </a>
  </TD></TR></TABLE>
</modality>
<modality class="speech" module="vxml">
  <block>
    Shop CNN for all your information needs!
  </block>
</modality>
<menu name="cnn.command"
  node_id="2">
  <choices node_id="3" >
```

```

        <default value="#cnn.query">Select News Stories</default>
        <choice value="#cnn.exit"
            require_confirmation="true">
            Exit </choice>
    <choice value="#cnn.applicationHelp">Help</choice>
</choices>
</menu>
<cml name="cnn.applicationHelp"
    title="About CNN Mobile"
    node_id="4"
    action="return">
<message
    node id="5" >
    This application allows you to select and view CNN news stories
</message>
</cml>
<cml name="cnn.exit"
    node_id="6"
    title="Exit CNN Mobile News"
    action="submit">
<message node_id="60">
    Thankyou for using the CNN news service
</message>
</cml>
<group node_id="7"
    groupId="query">
    <cml name="cnn.query"
        title="Search CNN Mobile News"
        node_id="58">
    <menu name="cnn.query.topic"
        node_id="11"
        title="Topic Selection">
    <choices node_id="12" >
        <choice value="#cnn.query.news"> News </choice>
        <choice value="#cnn.query.business"> Business </choice>
        <choice value="#cnn.query.sports">
            <grammar> (sport | sports" </grammar>
            Sports
        </choice>
        <choice value="#cnn.query.travel"> Travel </choice>
        <choice value="#cnn.query.weather"> Weather </choice>
        <choice value="#cnn.query.show">
            <grammar > show [business] </grammar>
            Show business
        </choice>
    </choices>
    </menu>
</cml>
<cml name="cnn.query.news"
    title="News Channel"

```

```

        node_id="#13"
        action="submit">
<select name="cnn.query.part">
    <message node_id="#9" >
        Which part of today's news would you like to read?</message>
    <choices
        node_id="10" >
        <choice value="h"> Headlines</choice>
        <choice value="1"> first story </choice>
        <choice value="2"> second story </choice>
        <choice value="3"> third story </choice>
    </choices>
</select>
<select name="cnn.query.interest">
    <message node_id="14">
        Which news category would you like to read?
    </message>
    <choices node_id="15" >
        <choice value="business">
            <grammar type="text/jsf">
                business {BIZ}</grammar>
            Business
        </choice>
        <choice value="africa">
            Africa</choice>
        <choice value="world"> World </choice>
        <choice value="United states"> United states </choice>
            <choice value="europe"> Europe </choice>
        <choice value="Asia"> Asia</choice>
        <choice value="me"> Middle East</choice>
        <choice value="america"> America </choice>
    </choices>
</select>
</cml>
<cml name="cnn.query business"
    title="Business Channel"
    action="submit"
    node_id="16" >
    <select name="cnn.query.part">
        <message node_id="9" >
            Which part of today*s news would you like to read?</message>
        <choices
            node_id="10" >
            <choice value="h"> Headlines</choice>
            <choice value="1"> first story </choice>
            <choice value="2"> second story </choice>
            <choice value="3"> third story </choice>
        </choices>
    </select>
    <select name="cnn.query.interest">

```

```

<message node_id="17">
  Which business category would you like to read?</message>
<choices node_id="18">
  <choice value="NEWS"> news </choice>
  <choice value="IN"> indexes </choice>
  <choice value="CU"> exchange rates </choice>
  <choice value="MET"> metals </choice>
</choices>
</select>
</cml>
<cml name="cnn.query.weather"
  title="Weather Channel"
  action="submit"
  node_id="19" >
<select name="cnn.query.part">
  <message node_id="9" >
    Which part of today*s news would you like to read?</message>
  <choices node_id="10" >
    <choice value="h"> Headlines</choice>
    <choice value="1"> first story </choice>
    <choice value="2"> second story </choice>
    <choice value="3"> third story </choice>
  </choices>
</select>
<select name="cnn.query.interest">
<message node_id="20">
  Which region are you interested in?</message>
<choices node_id="21">
  <choice value="us"> United states </choice>
  <choice value="europe">
    <grammar type="text/jsgr"> (euro | Europe) </grammar>
    Europe
  </choice>
  <choice value="JP"> Japan </choice>
  <choice value="AU"> Australia </choice>
  <choice value="AS"> Asia </choice>
</choices>
</select>
</cml>
<cml name="cnn.query.travel"
  title="Travel Section"
  action="submit"
  node_id="22" >
<select name="cnn.query.part">
  <message node_id="9" >
    Which part of today*s news would you like to read?</message>
  <choices
    node_id="10">
    <choice value="h"> Headlines</choice>
    <choice value="1"> first story </choice>

```

```

        <choice value="2"> second story </choice>
        <choice value="3"> third story </choice>
    </choices>
</select>
<select name="cnn.query.interest">
    <message node_id="23">
        Which city do you want to visit?</message>
    <choices node_id="24">
        <choice value="AMSTERDAM">AMSTERDAM</choice>
        <choice value="COPENHAGEN">COPENHAGEN</choice>
        <choice value="HELSINKI">HELSINKI</choice>
        <choice value="HONGKONG">HONGKONG</choice>
        <choice value="LONDON">LONDON</choice>
        <choice value="OSLO">OSLO</choice>
        <choice value="PRAGUE">PRAGUE</choice>
        <choice value="SINGAPORE">SINGAPORE</choice>
        <choice value="STOCKHOLM">STOCKHOLM</choice>
        <choice value="SYDNEY">SYDNEY</choice>
    </choices>
</select>
</cml>
<cml name="cnn.query.sports"
    action="submit"
    title="Sports Channel"
    node_id="25">
    <select name="cnn.query.part">
    <message node_id="9">
        Which part of today's news would you like to read?</message>
    <choices
        node_id="10" >
        <choice value="h"> Headlines</choice>
        <choice value="1"> first story </choice>
        <choice value="2"> second story </choice>
        <choice value="3"> third story </choice>
    </choices>
    </select>
    <select name="cnn.query.interest">
        <message node_id="26">
            What sports are you interested in?</message>
        <choices node_id="27">
            <choice value="AS"> Asia </choice>
            <choice value="w"> world </choice>
            <choice value="eu"> europe </choice>
            <choice value="us"> united states </choice>
            <choice value="nba"> NBA </choice>
            <choice value="nhl"> nhl </choice>
            <choice value="EF"> European football </choice>
        </choices>
    </select>
</cml>

```

```

<submit target="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi">
  <message node_id="28">
    executing <value name="cnn.command"/>
    for <value name="cnn.query.part"/>
    stories about <value name="cnn.query.interest"/>
    from topic <value name="cnn.query.topic"/>
  </message>
  <env name="cnn.command"/>
  <env name="cnn.query.topic"/>
  <env name="cnn.query.interest"/>
  <env name="cnn.query.part"/>
</submit>
</group>
<submit target="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi">
</submit>
</cml>

```

【0154】以下に、(CMLからHTMLへの)XSLソースをHTMLによって表面変更されたCMLソース・ページ上に適用することによって得られるHTMLソース・ページを記述する。得られた歓迎GUIページをHTMLブラウザで見たものを、図11に示す。表面変更は、表面変更されないページと比較するときにはつ

きりと見る事ができる。これは、意のままにページを表面変更する可能性を示す。この場合もやはり、すべてのケースを考慮しているわけではないが、これは、この手法を明瞭に示している。

【0155】以下は、得られた表面変更済みのHTMLソース・ページに関連するコードである。

```

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<META http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<title>CNN Mobile News</title>
</head>
<body>
<a name="cnn">CNN Mobile News</a>
</h1>
<div node_id="1" name="cnn">
  <LINK REL="stylesheet" HREF="cnn.css" TYPE="text/css">
  <TABLE BORDER="0" WIDTH="600" CELSPACING="0" CELLPADDING="0">
<TR>
<TD WIDTH="122" VALIGN="TOP"><a HREF="http://cgi.cnn.com/cgi-bin/redir ect?cnn_store">
<IMG SRC="http://cnn.com/images/1999/10/cnnstore.gif" WIDTH="120" HEIGHT="60" BORDER="1" ALT="CNN Store"></a></TD>
<TD WIDTH="8" VALIGN="TOP"><a HREF="http://cnn.com/ads/e.market/">
  <IMG SRC="http://cnn.com/images/1998/05/homepage/ad.info.gif" WIDTH="7" HEIGHT="62" BORDER="0" ALT="ad info"></a></TD>
<TD WIDTH="470" VALIGN="TOP">
  <a HREF="http://cnn.com/event.ng/Type=click%26RunID=11875%26 ProfileID=34%26AdID=13042%26GroupID=15%26FamilyID=1099%26TagValues=4.8.249.435.594.606%26Redirect=http:%2F%2Fwww.cnn.com%2FHLN%2Findex_pgm.htm" target="_top">
    <img src="http://cnn.com/ads/advertiser/promo/inter_company_

```



onair/9907/onair\_egg\_cnn.gif" border="0" height="60" width="468" alt="Get to the point news!">

</a>

<table width="100%" cellpadding="0" cellspacing="0" border="0">

<tr>

<td align="right"><font face="verdana, ARIAL, sans-serif" size="1"><a href="http://cnn.com/event.ng/Type=click%26RunID=11875%26ProfileID=34%26AdID=13042%26GroupID=15%26FamilyID=1099%26TagValues=4.8.249.435.594.60 6%26Redirect=http:%2F%2Fwww.cnn.com%2FHLN%2Findex\_pgm.htm target=\_top">Get to the point news!</a></font></td>

</tr>

</table>

</TD>

</TR>

</TABLE>

<ol node\_id="2">

<li>

<a href="#cnn.query">Select News Stories</a>

</li>

<li>

<a href="#cnn.exit">

Exit </a>

</li>

<li>

<a href="#cnn.applicationHelp">Help</a>

</li>

</ol>

<h2 node\_id="4">

<a name="cnn.applicationHelp">About CNN Mobile</a>

</h2>

<P node\_id="5">

This application allows you to select and view CNN news stories

</P>

<p>

<a href="#cnn">

Back

</a>

</p>

<h2>

<a name="cnn.exit">Exit CNN Mobile News</a>

</h2>

<form node\_id="6" action="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi">

<P node\_id="60">

Thankyou for using the CNN news service

</P>

<p>

<INPUT TYPE="SUBMIT" VALUE="cnn.exit">

< /p>

```

</form>
  <div groupId="query" modality="" class="">
    <h2 node_id="8">
      <a name="cnn.query">Search CNN Mobile News</a>
    </h2>
    <h2>
      <a name="#cnn.query.topic">Topic Selection</a>
    </h2>
    <ol node_id="11">
      <li>
        <a href="#cnn.query.news"> News </a>
      </li>
      <li>
        <a href="#cnn.query.business"> Business </a>
      </li>
      <li>
        <a href="#cnn.query.sports">
          Sports
        </a>
      </li>
      <li>
        <a href="#cnn.query.travel"> Travel </a>
      </li>
      <li>
        <a href="#cnn.query.weather"> Weather </a>
      </li>
      <li>
        <a href="#cnn.query.show">
          Show Business
        </a>
      </li>
    </ol>
    <h2>
      <a name="cnn.query.news">News Channel</a>
    </h2>
    <form node_id="13" action="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi">
      <p node_id="9">
        Which part of today's news would you like to read?</P>
      <select name="cnn.query.part">
        <option value="h"> Headlines</option>
        <option value="1"> first story </option>
        <option value="2"> second story </option>
        <option value="3"> third story </option>
      </select>
    <p>
    < /p>
    <p node_id="14">
      Which news category would you like to read?
    </P>

```

```

<select name="cnn.query.interest">
    <option value="business">
        Business
    </option>
    <option value="africa">
        Africa</option>
    <option value="world"> World </option>
    <option value="United states"> United states </option>
    <option value="europe"> Europe </option>
    <option value="Asia"> Asia</option>
    <option value="me"> Middle East</option>
    <option value="america"> America </option>

</select>
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.news">
< /p>
</form>
    <h2>
<a name="cnn.query.business">Business Channel</a>
</h2>
<form node_id="16" action="http://raman.almaden.ibm.com/cgi-bin/
cnn.cgi">
    <P node_id="9">
Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
    <option value="h"> Headlines</option>
    <option value="1"> first story </option>
    <option value="2"> second story </option>
    <option value="3"> third story </option>
</select>

<p>
</p>
    <P node_id="17">
Which business category would you like to read?</P>
<select name="cnn.query.interest">
    <option value="NEWS"> news </option>
    <option value="IN"> indexes </option>
    <option value="CU"> exchange rates </option>
    <option value="MET"> metals </option>
</select>

<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.business">
< /p>
</form>
    <h2>

```

```

<a name="cnn.query.weather">Weather Channel</a>
</h2>
<form node_id="19" action="http://raman.almaden.ibm.com/cgi-bin/
cnn.cgi">
    <P node_id="9">
Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
    <option value="h"> Headlines</option>
    <option value="1"> first story </option>
    <option value="2"> second story </option>
    <option value="3"> third story </option>
</select>

<p>
</p>
    <P node_id="20">
        Which region are you interested in?</P>
<select name="cnn.query.interest">
    <option value="us"> United states </option>
    <option value="europe">

        Europe
    </option>
    <option value="JP"> Japan </option>
    <option value="AU"> Australia </option>
    <option value="AS"> Asia </option>
</select>

<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.weather">
</p>
</form>
    <h2>
<a name="cnn.query.travel">Travel Section</a>
</h2>
<form node_id="22" action="http://raman.almaden.ibm.com/cgi-bin/
cnn.cgi">
    <P node_id="9">
        Which part of today's news would you like to read?</P> <select
name="cnn.query.part">
            <option value="h"> Headlines</option>
            <option value="1"> first story </option>
            <option value="2"> second story </option>
            <option value="3"> third story </option>
        </select>

<p>
</p>
    <P node_id="23">
        Which city do you want to visit?</P>
<select name="cnn.query.interest">

```

```

        <option value="AMSTERDAM">AMSTERDAM</option>
        <option value="COPENHAGEN">COPENHAGEN</option>
        <option value="HELSINKI">HELSINKI</option>
        <option value="HONGKONG">HONGKONG</option>
        <option value="LONDON">LONDON</option>
        <option value="OSLO">OSLO</option>
        <option value="PRAGUE">PRAGUE</option>
        <option value="SINGAPORE">SINGAPORE</option>
        <option value="STOCKHOLM">STOCKHOLM</option>
        <option value="SYDNEY">SYDNEY</option>
    </select>
</p>
</p>
<p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.travel">
</p>
</form>
    <h2>
<a name="cnn.query.sports">Sports Channel</a>
</h2>
<form node_id="25" action="http://raman.almaden.ibm.com/cgi-bin/
cnn.cgi">
    <P node_id="9">
        Which part of today's news would you like to read?</P>
    <select name="cnn.query.part">
        <option value="h"> Headlines</option>
        <option value="1"> first story </option>
        <option value="2"> second story </option>
        <option value="3"> third story </option>
    </select>
</p>
</p>
    <P node_id="26">
        What sports are you interested in?</P>
    <select name="cnn.query.interest">
        <option value="AS"> Asia </option>
        <option value="w"> world </option>
        <option value="eu"> europe </option>
        <option value="us"> united states </option>
        <option value="nba"> NBA </ option>
        <option value="nhl"> nhl </option>
        <option value="EF"> Europoean football </option>
    </select>
</p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE=" cnn.query.sports">
</p>
</form>
</div>

```

```

</div>
</body>
</html>

```

【0156】N. CML DTD 文書型定義

以下は、CML DTDを表現する。以下のDTD記述が、XMLの技術に精通する者なら誰にでも完全に理解

されるべきであることを理解されたい。これは、この実施形態に対して呈示されるCMLの構文を完全に定義する。

```

<!--$Id: cml.dtd,v 1.14 2000/03/02 17:04:02$ -->
<!-- DTD For Conversational Markup Language CML -->
<!--Conventions:
Tags are all lower case.
Attribute names are all lower case. -->
<!-- {attribute entities -->
<!-- core attributes common to most elements
node_id document-wide unique id
name Names data item that is populated by this gesture.
title Human readable title
style URI of custom stylesheet
-->
<!ENTITY % coreattrs
"node_id ID #IMPLIED
name CDATA #IMPLIED
style CDATA ; #IMPLIED
trigger CDATA #IMPLIED
title CDATA #IMPLIED"
>
<!-- } -->
<!-- {entities -->
<!ENTITY % GESTURE "(cml
| select
| menu
| message
| help)">
<!-- } -->
<!-- { TOP LEVEL CML -->
<!ELEMENT group (
%GESTURE+)
>
<!ATTLIST group
id ID #required
modality CDATA #IMPLIED
class CDATA #IMPLIED
>
<!ELEMENT CML (
(group | %GESTURE)+,
submit?
)
>
<!ATTLIST cml %coreattr>
<!-- } -->
<!-- {gesture message

```

```

<!ELEMENT message ANY>
<!ATTLIST message %coreattr>
<!-- } -->
<!-- {gesture help
<!ELEMENT help ANY>
<!ATTLIST help %coreattr>
<!-- } -->
<!-- {gesture boolean
<!ELEMENT boolean(
message,
help?)
>
<!ATTLIST boolean %coreattr;
require_confirmation (true | false ) #implied
require_confirmation_if_yes (true | false #implied
require_confirmation_if_no (true | false #implied
default (true | false #implied
>
<!-- } -->
<!-- {gesture select
<!ELEMENT error ANY>
<!ELEMENT grammar (
gram,
help?)
>
<!ATTLIST grammar
type CDATA #required>
<!ELEMENT gram ANY>
<!ELEMENT final ANY>
<!-- open content model for element predicate for now -->
<!-- will use an expression syntax a la xpath and augmented -->
<!-- as needed -->
<!-- will also draw on xforms work -->
<!ELEMENT predicate ANY>
<!ELEMENT choice (
grammar?,
PCDATA)
>
<!ATTLIST choice %coreattr;
value CDATA #required
>
<!-- default has same content model as choice -->
<!ELEMENT default (
grammar?,
PCDATA)
>
<!ATTLIST default %coreattr;
value CDATA #required
>
<!ELEMENT choices (

```

```

choice+,
default?)
>
<!ELEMENT select (
message,
help?,
choices,
predicate?,
error?)
>
<!ATTLIST select %coreattr;
require_predicate (true | false ) #IMPLIED
selection_type CDATA #IMPLIED
>
<!-- } -->
<!-- {gesture menu
<!ELEMENT menu (
message,
help?,
choices)
>
<!ATTLIST menu %coreattr; >
<!-- } -->
<!-- {constrained input -->
<!-- CML provides gestures for standard dialog components,
the following is merely a sample list of gestures:
Date
Specify date
Time
Specify time.
Currency
Specify currency amount.
Credit card
Specify a credit card (including card type, card number and
expiration date).
Phone
Specify a telephone number.
Email
Specify an email address.
url
Specify a url.
Snail Address
Specify a snail mail address, including street, city/state/country
and zip code.
We will specify formal DTD for these elements. -->
<!-- } -->
<!-- {unconstrained input -->
<!ELEMENT input (
message,
help?,

```



```

predicate?)
>
<!-- input %coreattr;
require_predicate (true | false ) #implied
>
<!-- } -->
<!-- {gesture user_identification
<!-- ELEMENT user_identification (
message,
help?,
user,
identify,
predicate,
error)
>
<!-- user_identification %coreattr;
require_predicate (true | false ) #implied
on_fail CDATA #implied
>
<!-- } -->
<!-- {gesture submit -->
<!-- ELEMENT env EMPTY>
<!-- ATTLIST env
name CDATA #required>
<!-- ELEMENT submit (
message?,
help?,
env*)
>
<!-- submit
target CDATA #required>
<!-- } -->
<!-- {binding events -->
<!-- ELEMENT bind-event EMPTY>
<!-- ATTLIST bind-event
logical CDATA #required
physical CDATA #implied
modality CDATA #implied
>
<!-- } -->
<!-- {environment
<!-- ELEMENT var EMPTY>
<!-- ATTLIST var
name CDATA #required
value CDATA #implied
>
<!-- ELEMENT value EMPTY>
<!-- ATTLIST var
name CDATA #required
>

```

```

<!ELEMENT assign EMPTY>
<!ATTLIST var
  name CDATA #required
  value CDATA #required
>
<!-- } -->
<!-- { end of file -->
<!-- End Of DTD
local variables:
folded-file: t
end:
-->
<!-- } -->

```

【0157】したがって、本発明による会話型マークアップ言語は、本明細書に詳細に述べるように、多くの有用な特徴および利点を提供する。対話によるプログラミングは、実装されている、基礎をなすデータ・モデルの定義（モデル）が、ユーザ対話を定義するマークアップ言語（ビュー／コントローラ）から分離されることを可能にする。これにより、密に同期化されたマルチモーダル対話の構築が可能となり、会話型アプリケーションがサポートされる。本発明によるCMLは、密な同期化をサポートする機構、すなわち各ジェスチャにNode\_id属性を結合し、この属性を様々な出力にマッピングすることを提供する。この言語は、原子構成（ジェスチャ）で定義されることが好ましく、より複雑な構成がもしあれば、それらは、（ダイアログで表した）複雑な構成の明瞭な意味定義と共にこれらの原子ジェスチャで構成される。これにより、複雑なモジュールを異なるモーダル性にマッピングすることができる。音声は、GUIと同じレベルにある第1のクラスのユーザ・インタフェース（UI）モーダル性と考えられる。ジェスチャは、基本のダイアログ・コンポーネントに対応する（これは、適切なデータ・ファイルを追加することを含む）。必要とされる場合に、モーダル性特定のコンポーネントをカプセル化したいと思う作成者は、モーダル性特定のマークアップをコード化するための「通過」機構を提供することができる。モーダル性特定の構造（スピーチ用またはGUI用）は、この通過機構に限定される。会話型UIがサポートされる。マークアップ言語は、並列で活動化できるダイアログ・コンポーネントを取り込む。CMLは拡張可能言語であり、例えば、新しいジェスチャを定義することができ、ジェスチャ変換規則を修正することができ、他の言語からのタグ／構成を埋め込むことができる（通過モードで）。モーダル性特定のタグ／通過は、ページを追加で表面変更するための機構でしかない。CMLはまた、アプリケーション状態をカプセル化するための明示的な環境も提供する。CMLはまた、動的に生成されたデータに言及する対話記述ならびにバックエンドへのサポート用コールバック機構の能力も提供

する。どんな従来方法も、これらの目的に使用することができる。さらに、本明細書に提供するCMLの詳細な説明が与えられれば、当業者は、この創意に富んだマークアップ言語の使用に関連する様々なツールおよび開発環境を実現することができる。

#### 【0158】II. マルチモーダル・ブラウザ

以下は、本発明によるマルチモーダル・ブラウザの説明である。このセクションは、参照しやすいように次のサブセクションに分かれている。すなわち、（A）序説、（B）マルチモーダル・シェル、（C）マルチモーダル・シェルとCML、（D）CMLとマルチモーダル同期化、（E）CMLとアプリケーション・オーサリング、（F）例示的な実施形態、（G）代替実施形態である。

#### 【0159】A. 序説

本発明によるマルチモーダル・ブラウジングを説明する前に、本発明のCMLおよびマルチモーダル・ブラウザに関係する概念を有する、上に参照した特許出願のいくつかの記述を要約したものを以下に述べる。参照しやすいように、関連出願は、それらの各整理番号を介して言及する。

【0160】YO999-111は、会話型コンピューティング、会話型ユーザ・インタフェース、および会話型アプリケーション・プラットフォーム（CVM、会話型仮想マシン）の概念を開示する。YO999-111に述べられている、CVMから提供される機能および挙動／サービスは、実際には、本発明のマルチモーダル・ブラウザによって、または会話型ユーザ・インタフェースを提供するアプリケーションによって実施される。しかし、概念レベルでは、CVMが、本発明のブラウザをサポートするのに必要なすべてのサービスを実施すると仮定される。

【0161】YO998-392は、会話型アプリケーション（すなわちマルチモーダル）をプログラムするための宣言的なプログラミング言語（「CML」と言及されるが、これはこの場合、本発明の言語とは異なる）の使用を開示する。YO998-392言語は、マルチモーダル／会話型ユーザ・インタフェースをサポートする

宣言的な言語である。実際には、その明細書に提供されている例／実施形態は、本発明によって可能な単一のオーサリングではなく「複数オーサリング」モデルに従って書かれたMLページからなる。以下の通り、宣言的なプログラミング言語の異なる例が教示された。

( i ) スピーチのみのML。スピーチMLとも呼ばれ、VoiceXMLに通じる。

( ii ) ファイル間の同期化タグを有する複数のファイル ( HTMLとVoiceXML、またはWMLとVoiceXML ) 。

( iii ) 複数のモーダル性記述を有する単一のファイル ( 例えば<MM><Speech>Speech rendering info</speech><GUI>GUI rendering info</GUI></MM>など ) 。この場合も同期化情報を有する。

( iv ) 異なるモーダル性に関連する情報を分割するためのフレーム様モデルを有する単一のファイル ( 例えば、スピーチ・コンテンツがHTMLページに加えて「スピーチ・フレーム」中に呈示される ) 。

【0162】これらのアイテムはどれも、単一オーサリングに対処しない。また、CML、任意のターゲット・レガシーML ( チャンネル ) 、あるいはジェスチャまたはジェスチャ・ベースのXSLの概念からのサポートにも対処しない。

【0163】YO999-178は、一般マルチモーダル・シェルを記述する。これは、同期化されるマルチモーダル・アプリケーション ( 宣言型でも命令型でもハイブリッドでも ) をどのようにサポートおよびプログラムするかを記述する。これは、各アプリケーション・モーダル性がある状態、それがサポートするコマンド、およびそれらのコマンドが他のモーダル性に与える影響を登録する登録テーブルを使用する。この場合もまた、ジェスチャおよび単一オーサリングの教示はない。一実施形態は、アプリケーションがブラウザ ( すなわち各モーダル性のレンダリングに関連するブラウザ ) であり、シェルがCMLページを受け取り ( YO998-392に定義されるように ) 、登録テーブルを構築し、したがってモーダル性にわたって同期をとる体系を記述する。

【0164】ここで本発明は、以下の記述に説明するように、マルチモーダル・ブラウザ体系を可能にする。このようなマルチモーダル・ブラウザは、以下に述べるが、上のセクションIで詳細に述べたCMLおよびその言語の会話型ジェスチャの特徴および利点を利用して、ユーザがアプリケーションによってサポートされる任意の装置上にある任意のモーダル性の情報にアクセスできるようにする。例えば、マルチモーダル・ブラウザ内の視覚的および話される対話は、会話型ジェスチャのコア・セットを使用して抽象され、CMLを使用して表現される。対話型ジェスチャは、各対話モーダル性によって適切に実現される。これらの基本的な会話型ジェスチャを使用して軽量情報アプリケーション ( インフォウエ

ア ) をオーサリングすることができ、得られたコンテンツは、表現されるときに、モーダル性／装置特定のマークアップ言語またはワイヤ・プロトコル ( wire protocol ) 、例えばいくつかを挙げればVoiceXML、WMLに投影される。

【0165】B. マルチモーダル・シェル  
マルチモーダル・ブラウザの動作の中心には、マルチモーダル・シェル機構がある。マルチモーダル・シェルは、複数のユーザ・インタフェース・クライアントまたはブラウザに対するサーバの役割を果たす。異なる対話モーダル性を備えるブラウザ、例えば視覚HTMLブラウザまたは聴覚VoiceXMLブラウザは、マルチモーダル・シェルにクライアントとして登録する。ユーザ対話は、マルチモーダル・シェルがCML文書を横断することによって進行する。この横断中に、シェルは、以下によって、ユーザの対話を特定の個々のCMLインフォウェアに対して調整する。

( i ) 現行のCMLノードの対話特定の表現を、すべての登録済みクライアントに渡すことによってユーザ対話を開始する。

( ii ) 現行のCMLノードを受け取ったすべての登録済みクライアントからの情報更新を待機する。

( iii ) 受け取った情報間の矛盾、例えばユーザが右と言っているのに左を指しているなどをおそらく解消する。

( iv ) 受け取ったばかりの情報更新に基づいて現行のCMLノードを更新する。

( v ) 更新の実行が成功すると、シェルは新たに更新されたアプリケーション状態をすべての登録済みブラウザに渡す。

【0166】C. マルチモーダル・シェルとCML  
上に説明したように、CMLアプリケーションは、標準的な会話型ジェスチャのセットの集合である。このような会話型ジェスチャは、アプリケーションを構成する完全なダイアログの基本的な構成単位を形成する。例えば、特定のアプリケーションでは、アプリケーション設計者の主要な作業は、以下のものを指定することである。

( i ) ユーザから収集する情報のアイテムを指定する。

( ii ) 必要なアイテムごとに制約、例えばセットからの選択などを指定する。

( iii ) 情報の各アイテムに必要なものが備わったときに、アプリケーション状態を更新する。

( iv ) 収集した情報のアイテムをパッケージして、バックエンド・アプリケーション・サーバにサブミットする。

【0167】上の作業は、指定されるとき、使用中の対話モーダル性に対して独立であることに留意されたい。

【0168】異なるユーザ・インタフェース・フロントエンド、例えば視覚WWWブラウザ、聴覚VoiceXML

などは、これらの作業を適切なユーザ・インタフェースの仕掛けにマッピングする。

【0169】CML文書は、一般マルチモーダル・シェルによってホスティングされる。シェルは、異なるユーザ・インタフェースの実現、例えば、視覚HTMLブラウザまたは聴覚VoiceXMLブラウザを助ける。シェルのクライアントになりたいと思うブラウザは、現行のアプリケーション状態への弱い参照を保持する。登録済みクライアントは、アプリケーション状態が変化したときにシェルから通知され、次いで各クライアントは、それ自体の弱い参照をアプリケーション状態に照会して、ユーザに呈示したいと思う関係情報を抽出する。

【0170】ユーザは、登録済みブラウザの1つを介してアプリケーションと対話することによって、CML文書を横断する。対話が進行するにつれて、すべての登録されたブラウザは、対話のフォーカスである現行のCMLノードについて通知され、したがって、それらの表現を必要に応じて更新する。シェルは、現在開かれているCML文書、ならびにそれらに対応するアプリケーション状態を常に監視する。必要な場合に、会話型シェルは、現在開かれているアプリケーションのいずれかの状態の簡潔な要約を提供することができる。登録済みクライアントのいずれかを介してサブミットされた情報はシェルによって媒介され、シェルは、他の登録済みクライアントへの、かつ必要な場合にバックエンド・アプリケーション・サーバへの通知を扱う。

【0171】D. CMLおよびマルチモーダル同期化  
単一のCML表現からのアプリケーションを対話特定に具現化したもの総合することによって、マルチモーダル・インタフェースの異なる態様を同期させることができる。CML表現中の各ノードは、特定のnode-idでタグ付けされる。CML表現が対話特定の表現、すなわちHTMLまたはVoiceXMLにマッピングされるとき、得られたマッピングにおけるノードは、それらに対応するCML表現中のノードのnode-idでタグ付けされる。ユーザが特定のモーダル性を介してブラウザと対話するとき、マルチモーダル・シェルは、関係するnode-idを検索することによって、アプリケーション中の現在活動化されているノードを元のCML表現にマッピングし返す。ユーザ対話のせいでアプリケーション状態が変化するとき、シェルは、修正されたアプリケーション状態を、修正されたノードのnode-idと共に、通知されるように登録されたすべてのクライアントに渡す。通知されたアプリケーションは、node-idに対してチェックすることによって、それらの対話特定の表現中の対応するノードを更新する。登録済みのアプリケーションは本質的に、基礎をなすアプリケーション状態への弱い参照を保持することに留意されたい。対話特定のレンダリング・エンジンが必要なノードを更新するとき、弱い参照は、その更新に関係する情報（かつ、必要とされる情報だ

け）が自動的にシェルから検索されるようにすることになる。

【0172】ここで図12を参照すると、MVCモデルの新しい解釈が示されている。この新しい解釈によれば、このモデルは対話のCML記述である。ビューは、ジェスチャ・ベースのXSL変換規則を適用して、異なるレンダリング・ブラウザ中にレンダリングされる（ビュー）異なるターゲットMLを生成することによって得られるものである。ブラウザは、ユーザとの対話を通してモデルの制御を提供する（かつ、I/Oイベントがレンダリング・ブラウザの1つで発生したときにその状態を修正する）。図9に従って、V0がGUIビュー（例えばHTML）でありV1がスピーチ・ビュー（自然言語による、またはそうでない）であると想像されたい。C0は、モノモーダルHTMLブラウザのみの制御／対話である。C1は、同期化されるマルチモーダル・ビューである。C2は、モノモーダル・スピーチ制御である。この手法は、根本的に新しいパラダイムである。

【0173】E. CMLとアプリケーション・オーサリング

アプリケーション・クリエータは、WYSIWYG（what you see is what you get）オーサリング・ツールと対話して、自分のアプリケーションのCML表現を作成することができる。CMLで表現されたアプリケーションは、スタイル変換の標準セットを使用して、対話特定の表現、例えばVoiceXMLまたはHTMLにマッピングされる。必要とされる場合に、ユーザ・インタフェース設計者は、カスタム・スタイル変換を生み出して、特定のルック・アンド・フィールまたはサウンド・アンド・フィールを設計することができる。CMLオーサリング・ツールを生み出すこともでき、これは、クライアントがマルチモーダル・ブラウザ・プラットフォーム上で展開するためにレガシーHTMLのみのWWWアプリケーションをCMLにマッピングすることができるように作成することができる。このようなツールは、顧客が既存のWWWアプリケーションをVoiceXMLプラットフォーム上で展開することを助けるのに必要な橋を提供する。この解決法は、VoiceXMLに直接オーサリングし直すよりも魅力がある。というのは、既存のアプリケーションをCMLに一度マッピングすることは、様々なマルチモーダル・ブラウザ設定にわたる展開を可能にするからである。これは、HTML、WML（およびその他のレガシーML）にも当てはまる。

【0174】F. 例示的な実施形態

次に図13～15を参照すると、既存のシステムから本発明によってマルチモーダル・ブラウジング環境でCMLを完全に使用するまでの移行ロード・マップが示されている。

【0175】図13に、現行のファット・クライアント・ウェブ・プログラミング・モデルを示す。コンテンツ

は、主にHTMLで書かれる（そのフォーマットで静的に記憶される、または動的に生成される）。コンテンツが特定のブラウザ（例えば所与のバージョンのInternet explorerまたはCommunicator）に適合される必要があるとき、ターゲット・ブラウザの機能である特定のスタイル・シート、ならびにコンテンツのタイプが構築される。これは通常、XML/XSLオーサリング手法である。別のチャネル／モーダル性（WML、CHTML、VoiceXMLなど）が必要な場合、コンテンツは、書直す必要があるか、あるいはHTMLまたはXMLで書かれた場合には非常に特定の規則に従って、よく知られたタイプ／領域のものとする必要があり、したがって、いくつかの一般アプリケーション／ビジネス論理に依存するXSL規則を使用してこれらのモーダル性特定のレガシー言語を作成することができるか、またはXSL規則を非常に頻繁にオーサリングし直さなければならないか、あるいはその両方である。これは、異なるレガシー言語で直接にオーサリングしようが、単一のXMLコンテンツをこれらの異なるレガシーMLに変換する異なるスタイル・シートでオーサリングしようが、過大な複数オーサリングとなる。最終的に、今日、ウェブ（すなわち主にHTMLの交換による）、無線ネットワーク（主にWML、ただし他の規格も存在する）、および電話（主にVoiceXML）にアクセスする必要が一層多くなっている。複数オーサリングが唯一の解決法のため、このようなタイプのサービスを提供するサイトは、通常、限られた量のサービス／コンテンツ・プロバイダまたは企業サイトによる、閉じたサイト（開いた完全なウェブ・コンテンツとは反対に、限られた量のサービス／コンテンツ）である。いつでもどこでも、どんな情報へのアクセスをどんなアクセス装置を介しても提供し、ユーザがそれを操作できるようにする解決法は存在しない。異なるレガシー言語（XMLを含む）は、ページの異なる部分を適切に他のモーダル性で扱うのに必要な情報を含まない（例えば会話エンジン用の文法およびその他の引数が欠落しているなど）。

【0176】図14に、CMLを展開して、対話プログラミング・モデルおよび会話型コンピューティング・パラダイムによるプログラミングを使用する第1のステップを記述する。この解決法は、トランスポート・プロトコルおよびネットワーク（例えばテレフォニーPSTN、無線ネットワーク（音声またはデータあるいはその両方）、voice over IP、TCP/IP-HTTP、WAPなど）、ならびにレガシー・ブラウザ（例えばHTMLブラウザ、WMLブラウザ、VoiceXMLブラウザなど）で表される、今日存在するインフラストラクチャを使用することができる。コンテンツがCMLで利用可能な場合、それは、静的に生成されようが動的に生成されようがページが提供されるとき、要求するブラウザによってサポートされるターゲット・レガシーMLに実

行中にトランスコーディングすることができる。ターゲットMLの決定は、ブラウザのタイプ、またはゲートウェイ、ブラウザ、サーバのIPに基づく。WAPゲートウェイはWMLページを受け取り、ブラウザは、（httpヘッダ中の）記述子またはアクセス機構に基づいてその要件を記述する（例えば、httpは、いくつかのCMLブラウザが利用可能になるまでの少なくとも展開の最初では、HTMLを暗黙指定することになる）。決定はまた、要求されたページに基づいても行われる。ブラウザがxxxx.htmlを要求する場合、それは、CMLがHTMLにトランスコーディングされることを意味し、yyyy.vxmlを要求する場合、それは、VoiceXMLにトランスコーディングされることを意味する、などである。明らかにこれは、現在のインフラストラクチャおよびその何らかの未来の進化形をサポートすることを保証する。

【0177】CMLブラウザ（すなわち会話型／マルチモーダル）が開放されるとき、それは、CMLページ（すなわちzzzz.cml）を要求することになり、また、それ自体をCMLブラウザとして記述することもできる。このような場合、ページはどんなトランスコーディングも必要とせずに提供される。これは、レガシー／今日のインフラストラクチャからCML／会話型支配のウェブ・プログラミング・パラダイムへのスムーズな移行を保証する。ここで、レガシー・コンテンツ（すなわちHTML、WML、VoiceXML、またはその他のレガシー言語、あるいはそれらすべてで書かれた静的または動的コンテンツ）は、CMLに変換される必要がある。ツールは、せいぜいCMLターゲットを「当てる」ことにしか使用できず、次いでそのCMLターゲットは、手作業で妥当性検査して再編集する必要がある。しかし、上に説明したのと同じ理由で、元のページが特定の規則に従って構築された場合、またはXMLタグが明確であり（領域特定であり）、したがってページ中のそれらの役割が明確である場合、実行可能な自動トランスコーディング・システムを使用することができる。

【0178】図15に、CML会話型（マルチモーダル・ブラウザ）が標準となったときの展開ロード・マップにおける次のステップを示す。したがって、今やトランスコーディングはブラウザの一部であり、ページはCMLでオーサリングされ提供される。レガシー（すなわち非CML）ページが提供されるとき、それは、マルチモーダル・シェルによって取り出されるが、次いで、対応するモーダル性を扱う対応するレンダリング・ブラウザに直接に送信されることになる。

【0179】CMLコンテンツおよびレガシー・コンテンツは、当然、依然として前述のようにCMLにオーサリングまたは変換される必要がある。

【0180】図16を参照すると、本発明によるマルチモーダル・ブラウザ体系のブロック図が示されている。図示のように、マルチモーダル・ブラウザ60は、マル

チモーダル・シェルまたは会話型シェル62、GUIレンダリング・ブラウザ・コンポーネント64、およびスピーチ・レンダリング・ブラウザ・コンポーネント66を備える。マルチモーダル・シェルはまた、「仮想ブラウザ」とも呼ばれる。マルチモーダル・ブラウザ60は2つのモーダル性、視覚（ブラウザ・コンポーネント64）およびスピーチ（ブラウザ・コンポーネント66）の使用を示しているが、本発明がこれらのモーダル性に限定されないことを理解されたい。マルチモーダル・ブラウザ60は、一般に次のように動作する。アプリケーションにアクセスしようとするユーザは、マルチモーダル・ブラウザのすべてか一部が常駐するクライアント装置（例えばパーソナル・コンピュータ、ラップトップ・コンピュータ、携帯情報端末など）とインタフェースする。図16に示す一般的な場合では、ユーザはテキスト・インタフェースもしくはグラフィカル・インタフェース（GUI入力／出力）またはその両方を介してこれを行うことができ、あるいはインタフェースはスピーチ（オーディオ入力／出力）を介したものとすることができ、あるいはその両方ができる。図16にはマルチモーダル・ブラウザ60を1つのブロック中に示してあるが、マルチモーダル・ブラウザはクライアントとサーバのコンピュータ・システムを両方含めた複数の装置を介して実施できることを以下に説明する。

【0181】ユーザの要求に基づいて、マルチモーダル・ブラウザ60は適切なURLをコンテンツ・サーバ69に送り、コンテンツ・サーバ69もまた、所望の特定アプリケーションへのアクセスを要求するために、同じクライアント装置に常駐することのできる会話型エンジン68にサービスする。次いで、アプリケーションに関連するCMLコードが、コンテンツ・サーバ69からマルチモーダル・ブラウザ60にダウンロードされる。次いでマルチモーダル・ブラウザは、そのCMLコードに関連する会話型ジェスチャに基づいて、モーダル性特定のレンダリング（GUI表現またはスピーチ表現あるいはその両方）を生成する。したがってユーザは、これらの表現を介してブラウザ60と対話する。

【0182】次に（引き続き図16も参照しながら）図17を参照すると、本発明の一実施形態によるマルチモーダル・ブラウザの動作をより詳細に示す流れ図が示されている。アプリケーション開発者は、アプリケーション、例えばインフォウェアと呼ばれる軽量アプリケーションをCMLで書く。CMLでオーサリングされたインフォウェアは、複数のモーダル性特定のブラウザ・コンポーネント（例えば図16の視覚ブラウザ64とスピーチ・ブラウザ66）の間を媒介する会話型シェル（例えば図16のマルチモーダル・シェル62）によってホスティングされる。マルチモーダル・シェルは、CMLインタプリタまたはプロセッサと考えることができる。これを、ブロック70として図17に示す。ユーザ対話

はCMLインタプリタによって進行し、CMLインタプリタは、ダウンロードされたCMLコードに関連するCMLインスタンスをHTML（ブロック77）やVoiceXML（ブロック78）などの適切なモーダル性特定の言語にマッピングする。これらのモーダル性特定の表現は、そのアプリケーションに関連するモーダル性特定のバージョンのダイアログをレンダリングする。ブロック70中に示すように、ノード（A）および矢印（B）は、CMLの宣言型プログラムを表す。CMLプログラム中のジェスチャは各ノードで表され、矢印は、あり得る分岐点またはループによる対話／ダイアログの流れを表す。各ジェスチャはノードID（node\_id）で識別され、これは、異なる登録済みのモーダル性の中で同期をとるために、活動化されたジェスチャを適切に識別することを可能にする。node\_idはジェスチャを識別し、したがって、CMLブラウザ（すなわちマルチモーダル・シェルまたは仮想ブラウザ）は、それがダイアログ・フロー中のどこであるか、およびそこからどこに行くべきか（例えば、異なるモーダル性を更新する、またはサーバに変数を送って新しいCMLページを取り出す）を知る。

【0183】CMLからモーダル性特定の表現への変換77および78は、XSL変換規則（または前述の他の変換機構）によって支配される。これらのXSL規則は、モーダル性特定である。これらの変換は、XSL規則74および登録テーブル76に従って、プレゼンテーション生成ブロック72によって扱われる。登録テーブル76は、デフォルトのジェスチャXSL変換規則、ならびに拡張形、アプリケーション特定、装置特定、またはユーザ特定の特定規則のリポジトリである。CMLインスタンスを適切なモーダル性特定の表現にマッピングする過程で、XSL規則は、モーダル性特定のユーザ対話を実現するのに必要な情報を追加する。一例としては、要素selectをVoiceXMLに変換するとき、関係するXSL変換規則は、その会話型ジェスチャに有効な選択肢をカバーする文法の生成を扱う。

【0184】CMLインスタンスをHTMLなどのモーダル性特定の表現に変換する過程は、単一のCMLノードを出力表現中のノードの集合にマッピングすることになる。これらの様々な表現にわたって同期をとるのを助けるために、CML属性node\_idが、所与のCMLノードから得られた出力ノードのすべてに適用される。所与のCMLインスタンスが適切なモーダル性特定のXSL規則によって異なる表現、例えばHTMLおよびVoiceXMLにマッピングされるとき、出力におけるツリーの形は、様々なモーダル性の中で異なる見込みが高い。しかし、属性node\_idは、モーダル性特定の各表現から元のCMLノードへの概念上のバックリンクを提供することにより、これらの表現の間で同期をとることを可能にする。これを、図17のブロック70に図示する。

【0185】ユーザ対話が進行するにつれて、現行のCMLインスタンスによって環境中で定義される変数は、妥当性検査された値に束縛される。この束縛は、まず、モーダル性特定の表現（登録済みクライアント）77および78のうちの1つで発生する。モーダル性特定の表現は、更新された環境および完了したばかりのジェスチャのnode\_idを含む適切なメッセージをCMLインタープリタ（マルチモーダル・シェル）に送る。更新された束縛がCMLインタープリタに伝達されると、CMLインタープリタは、完了したばかりのジェスチャのnode\_idでモーダル性特定の表現すべてにメッセージを送る。モーダル性特定の表現は、このメッセージを受け取ると、まずCMLインタープリタにそれらの表現に影響する環境の部分を照会することによって、それらの表現を更新する。

【0186】図18に、本発明の一実施形態によりCMLマルチモーダル・ブラウザによって行われる異なるステップを示す。CMLページがブラウザによって取り出されると、ブラウザは、XMLパーサと同様にCMLコンテンツを解析する（ステップ90）。ブラウザは、対話の内部表現（すなわち、ページ中に記述された異なるジェスチャのグラフ／ツリー）およびノードIDを構築する。ブラウザは、ブラウザに記憶されたジェスチャXSL変換（またはJava BeansやJava Server Pagesのような他の変換機構）を使用して（ブロック98）、異なるMLページを構築し（ステップ96）、これは各レンダリング・ブラウザに送られる（ステップ100）。あるモーダル性のI/Oイベント時に、その影響が対話グラフ（すなわちY0999-178に記載のようにMMシェル登録テーブル（ブロック94）に記憶された）のレベルで検査される（ステップ92）。ジェスチャXSL変換規則は、アプリケーション開発者が上書きし、それらをどこでダウンロードすべきかを示すことができることに留意されたい。これらはまた、普通ならデフォルト挙動となるものから、ユーザ、アプリケーション、または装置のプリファレンスによって上書きすることもできる。新しいジェスチャも追加することができ、その場合、関連するXSL規則が提供されなければならない（例えばそれを得るためのURL）。

【0187】前述のように、本発明は、複数装置ブラウジング環境または分散ブラウジング環境を可能にする。CMLの性質およびそれが効果的に複数のブラウザを同期させる能力のせいで、アプリケーションの様々な部分を別個のコンピューティング装置上に常駐させて実行することができる。次いでユーザは、1つのアプリケーションにアクセスするときに複数の装置、例えばラップトップ・コンピュータおよびセルラー・ホンと同時に対話することができる。これは実際、異なるモーダル性でのブラウジングに限定されない。同じモーダル性（例えばGUIのみ）中でも、同じコンテンツを表現してこの表

現をモーダル性にわたって同期化する必要があるとき、例えば1つの装置上で画像を、別の装置上でビデオを、3つ目の装置上でテキストと背景を足したものを表示するときに、同じ原理を使用して、前もってそれがどんな装置かを記述することができる。別の例は、1つの装置上でテキストおよび画像を、別の装置上でアプレットを表示するなどである。もっと多くの例も容易に考えられる。これは、カスタマイズされたジェスチャまたはジェスチャXSL規則の使用を必要とすることになる。あるいは、これは、それを行うための（他のジェスチャおよびデフォルト・レンダリングを有する）別のマークアップを必要とすることになる。

【0188】次に図19を参照すると、このような分散ブラウジング環境が示されている。マルチモーダル・ブラウザ62、視覚ブラウザ64、スピーチ・ブラウザ66、会話型エンジン68、およびコンテンツ・サーバ69の機能および動作は、図16および17に関して上に述べたものと同じである。しかし、見れば分かるように、コンポーネントは複数のコンピューティング装置上に分散されている。例えば、マルチモーダル・ブラウザ62はサーバ80上に常駐し、ビジュアル・ブラウザ64はクライアント装置82上に常駐し、スピーチ・ブラウザはサーバ84上に常駐する。これらのクライアント装置およびサーバ装置は、WWW、ローカル・ネットワーク、または他の何らかの適したネットワークを介して通信することができる。ユーザはクライアント装置82に対してローカルでよく、サーバ80と82とはリモートに位置する。あるいは、すべてのまたはいくつかのコンピューティング装置を一所に配置することもできる。ユーザがクライアント装置82と直接に対話するので、オーディオ入力／出力機能86（例えばマイクロホンおよびスピーカ）が装置82に備わり、これらはサーバ84にあるスピーチ・ブラウザに接続される。見れば分かるように、CMLアプリケーションの、同期した同じ動作は、マルチモーダル・ブラウザの様々なコンポーネントが別個のコンピューティング装置上に位置していても達成することができる。

【0189】本発明の方法を実施するための前述の各クライアント装置およびサーバは、メモリおよびI/O装置に動作可能に結合されたプロセッサを備えることができることを理解されたい。本明細書で使用する用語「プロセッサ」は、例えばCPU（中央処理装置）を備えるものなど、どんな処理装置も含むものとするを理解されたい。本明細書で使用する用語「メモリ」は、例えばRAM、ROM、固定記憶装置（例えばハード・ドライブ）、取外し可能記憶装置（例えばディスクett）、フラッシュ・メモリなど、プロセッサまたはCPUに関連するメモリを含むものとする。さらに、本明細書で使用する用語「入力／出力装置」または「I/O装置」は、例えばデータを処理装置に入力するための1つまた



は複数の入力装置、例えばキーボード、マイクロホンなど、および処理装置に関連する結果を呈示するための1つまたは複数の出力装置、例えばCRTディスプレイ、スピーカなど、あるいはその両方を含むものとする。入力／出力装置は、モーダル性特定であり、したがって、他の装置を採用することもできる。また、「プロセッサ」は複数の処理装置を指すことができ、処理装置に関連する様々な要素は他の処理装置によって共用することも理解されたい。したがって、本明細書に述べる本発明の方法を実行するための命令またはコードを含むソフトウェア・コンポーネントは、1つまたは複数の関連する記憶装置（例えばROM、固定または取外し可能メモリ）に記憶することができ、利用される準備ができたときに部分的にまたは全体で（例えばRAMに）ロードしてCPUによって実行することができる。

#### 【0190】G. 代替実施形態

本発明の教示から自明に得られる可能な拡張の中には次のものがある。

【0191】(i) 上に考察した複数装置ブラウジング（ある所与のモーダル性でも）。

【0192】(ii) 複数地理サポート。いくつかのジェスチャ（例えば電話番号、アドレスなど）を、ローカルのフォーマットならびに言語に適合させることができる。これは、テキスト間変換システムに結合して、異なるXML規則を簡単に介して完全に自動的なローカライズ機構（select yes/Noがselect Oui/Nonになる）を実現することができる。あるいは、このような自動的なトランスコードがない場合に、システムを開発／ローカライズ用ツールの一部として使用して、ローカライズ／国際化、地理／地域の適合を高速化することもできる。

【0193】(iii) 会話型ファウンデーション・クラス（Conversational Foundation Class）。会話型ファウンデーション・クラスは、モーダル性独立の、かつ並列でおよび逐次的に実行してより複雑なダイアログを構築できる命令型ダイアログ・コンポーネントとして、Y0999-111に導入された。これらは、会話型アプリケーション・プラットフォーム（CVM、会話型仮想マシン）から提供されるサービスと組み合わせられて、プラットフォームが提供するこれらのファウンデーション・クラスのライブラリにロード／リンクすることによって、命令型会話型（マルチモーダル・アプリケーション）のプログラミングを可能にする。各CVMプラットフォームがこれを提供するので、アプリケーション開発者はそれらを利用することができ、装置によってサポートされるモーダル性内のレンダリングおよびそれらの同期化を心配しなくてよい。したがって、本明細書に提供したCML仕様中に宣言的に定義した各ジェスチャは、逐次的に（1つずつ）または並列で（複数のフォームが一度に活動化されるように複数の活動化される）実行できる命令型の実施形態を（例えばJavaで）有することが

できる。CFCでのプログラミングは、対話による命令型のプログラミングに相当する。すなわち、何らかの命令型のジェスチャを使用してそれにリンクし、それをバックエンドにフックし、従来のコードによってそのジェスチャを共に結合する。このコードまたはCFC引数中に、いくつかのモーダル性特定のカスタマイズを加えることもできる。次いで、プラットフォーム（同じレベルの機能を実行するCVMまたはブラウザ）が適切なモーダル性内のレンダリングと、ファウンデーション・クラス中にハード・コーディングされたモーダル性の間の適切な同期化とを扱えるようにする。一例は、すべてのファウンデーション・クラスがJava Classとして提供される場合であろう。これにより、対話モデルによるプログラミングをJavaアプレットまたはサーブレットに拡張することが可能である。

【0194】(iv) 対話によるハイブリッド・プログラミングは、宣言型と命令型の結合である。すなわち、CFCおよびCFCを使用して（かつ、よりタスク特定に）構築された他のオブジェクト、例えばJavaアプレットへの呼出しを伴うCMLページである。したがって、対話プログラミング・モデルによるプログラミングは、一般にすべてのプログラミング・モデルをカバーすると考えられる。

【0195】(v) スクリプティング。CMLは、再利用したいと思うどんなスクリプティング（<http://www.ecma.ch/stand/ecma-262.html>に定義されているECMAScriptなど）もマルチモーダル・シェルのスクリプティング言語として直接サポートすることができる。モーダル性特定のスクリプト（JavascriptやWMLスクリプトのような）は、モーダル性特定のスクリプティング言語として考えなければならない。CMLのECMAScriptがどのようにレガシー・ブラウザ用に変換されることになるかのより詳細な挙動を今日（すなわち我々が今日のインフラストラクチャを使用する場合のステップのために）定義することは可能だが、これらは、モーダル性特定として（すなわち画像のように）単純に扱うことができる。

【0196】まとめとして、本発明の構成に関して以下の事項を開示する。

【0197】(1) ユーザから1つまたは複数のコンピュータ・ベースの装置を介してアクセス可能なアプリケーションをプログラムする方法であって、前記アプリケーションにアクセスするために使用される前記1つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話を表現するステップを含み、前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立、さらに、コンポーネントごとにトランスコーディングされ



て、前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの1つまたは複数のモーダル性特定のレンダリングを生成する方法。

( 2 ) 前記アプリケーションの少なくとも一部がサーバから、クライアントの役割を果たす前記1つまたは複数のコンピュータ・ベースの装置のうちの少なくとも1つにダウンロードされるクライアント／サーバ構成において、前記サーバに常駐する前記コンテンツ／アプリケーション論理への接続を提供するように動作可能なコードを前記アプリケーション中に含めるステップをさらに含む、上記( 1 )に記載の方法。

( 3 ) 前記コンテンツ／アプリケーション論理接続コードが、前記アプリケーションに関連する1つまたは複数のデータ・モデル、属性制約、および妥当性検査規則のうちの少なくとも1つを表す、上記( 2 )に記載の方法。

( 4 ) 前記1つまたは複数のモーダル性特定のレンダリングが、前記アプリケーションの一部のスピーチ・ベースの表現を含む、上記( 1 )に記載の方法。

( 5 ) 前記スピーチ・ベースの表現がVoiceXMLに基づく、上記( 4 )に記載の方法。

( 6 ) 前記1つまたは複数のモーダル性特定のレンダリングが、前記アプリケーションの一部の視覚ベースの表現を含む、上記( 1 )に記載の方法。

( 7 ) 前記視覚ベースの表現がHTML、CHTML、WMLのうちの少なくとも1つに基づく、上記( 6 )に記載の方法。

( 8 ) 前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的に表現される、上記( 1 )に記載の方法。

( 9 ) 前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって命令的に表現される、上記( 1 )に記載の方法。

( 10 ) 前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的かつ命令的に表現される、上記( 1 )に記載の方法。

( 11 ) 前記対話ベースのプログラミング・コンポーネントが、前記ユーザと前記1つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する基本要素を含む、上記( 1 )に記載の方法。

( 12 ) 前記対話ベースのプログラミング・コンポーネントが複合要素を含み、前記複合要素は、前記ユーザと前記1つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する2つ以上の前記基本要素の集合体である、上記( 11 )に記載の方法。

( 13 ) 前記対話ベースのプログラミング・コンポーネントの1つが会話型ジェスチャを表す、上記( 1 )に記載の方法。

( 14 ) 前記会話型ジェスチャが、ユーザへの情報メッセージをカプセル化するジェスチャを含む、上記( 13 )に記載の方法。

( 15 ) 前記会話型ジェスチャが、コンテキスト・ヘルプ情報をカプセル化するジェスチャを含む、上記( 13 )に記載の方法。

( 16 ) 前記会話型ジェスチャが、別のジェスチャの完了が成功したときに行われるアクションをカプセル化するジェスチャを含む、上記( 13 )に記載の方法。

( 17 ) 前記会話型ジェスチャが、イエス／ノー・ベースの質問をカプセル化するジェスチャを含む、上記( 13 )に記載の方法。

( 18 ) 前記会話型ジェスチャが、ユーザが選択肢のセットから選択することを期待される場合のダイアログをカプセル化するジェスチャを含む、上記( 13 )に記載の方法。

( 19 ) 前記選択ジェスチャが前記選択肢のセットを表すサブ要素を含む、上記( 18 )に記載の方法。

( 20 ) 前記選択ジェスチャが、選択がパスすべきテストを表すサブ要素を含む、上記( 18 )に記載の方法。

( 21 ) 前記選択ジェスチャが、前記テストが不合格の場合に呈示すべきエラー・メッセージを表すサブ要素を含む、上記( 20 )に記載の方法。

( 22 ) 前記会話型ジェスチャが、所与の会話型ジェスチャの結果を妥当性検査するための規則をカプセル化するジェスチャを含む、上記( 13 )に記載の方法。

( 23 ) 前記会話型ジェスチャが、文法処理規則をカプセル化するジェスチャを含む、上記( 13 )に記載の方法。

( 24 ) 前記会話型ジェスチャが、ユーザが前記アプリケーションの各部分をナビゲートするのを助けるダイアログをカプセル化するジェスチャを含む、上記( 13 )に記載の方法。

( 25 ) 前記会話型ジェスチャが、少なくとも1つのユーザ・ログインおよび認証の情報を求める要求をカプセル化するジェスチャを含む、上記( 13 )に記載の方法。

( 26 ) 前記会話型ジェスチャが、制約付きのユーザ入力を求める要求をカプセル化するジェスチャを含む、上記( 13 )に記載の方法。

( 27 ) 前記会話型ジェスチャが、制約のないユーザ入力を求める要求をカプセル化するジェスチャを含む、上記( 13 )に記載の方法。

( 28 ) 前記会話型ジェスチャが、情報のサブミットを制御するジェスチャを含む、上記( 13 )に記載の方法。

( 29 ) 論理入力イベント、ならびに、前記論理入力イベントと定義された前記論理入力イベントをトリガする物理入力イベントとの間の関連を定義する機構を提供するステップをさらに含む、上記( 1 )に記載の方法。

( 30 ) 前記コンポーネントごとのトランスコーディングがXSL変換規則に従って行われる、上記( 1 )に記載の方法。

( 31 ) 前記コンポーネントごとのトランスコーディングがJava Beanに従って行われる、上記( 1 )に記載の方法。

( 32 ) 前記コンポーネントごとのトランスコーディングがJava Server Pageに従って行われる、上記( 1 )に記載の方法。

( 33 ) 前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが、前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記1つまたは複数のモーダル性特定のレンダリングを同期させることを可能にする、上記( 1 )に記載の方法。

( 34 ) 前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが自然言語理解環境をサポートする、上記( 1 )に記載の方法。

( 35 ) 前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記1つまたは複数のモーダル性特定のレンダリングに関連するプレゼンテーション・フィーチャを表面的に変更することを可能にするコードを含めるステップをさらに含む、上記( 1 )に記載の方法。

( 36 ) コンポーネントごとにトランスコーディングして前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記1つまたは複数のモーダル性特定のレンダリングを生成するための規則を変更することを可能にするコードを含めるステップをさらに含む、上記( 1 )に記載の方法。

( 37 ) 実装されている、基礎をなすデータ・モデルの定義が、前記ユーザ対話を定義するマークアップ言語から分離される、上記( 1 )に記載の方法。

( 38 ) node\_id属性が各コンポーネントに付加され、前記属性は様々な出力の上にマッピングされる、上記( 1 )に記載の方法。

( 39 ) モーダル性特定のマークアップ・コンポーネントをカプセル化する通過機構が作成者に提供される、上記( 1 )に記載の方法。

( 40 ) 前記コンポーネントが並列に活動化させることができる、上記( 1 )に記載の方法。

( 41 ) 前記プレゼンテーションおよびトランスコーディングが拡張可能である、上記( 1 )に記載の方法。

( 42 ) 前記アプリケーションの状態がカプセル化される、上記( 1 )に記載の方法。

( 43 ) 前記表現が動的に生成されるデータの参照を可能にし、前記コンテンツ／アプリケーション論理へのコールバック機構をサポートする、上記( 1 )に記載の方法。

( 44 ) 1つまたは複数のコンピュータ・ベースの装置

に関連するアプリケーションにアクセスする際に使用する装置であって、1つまたは複数のプロセッサを備え、前記プロセッサが、( i ) アプリケーション・サーバから前記アプリケーションを得るように動作可能であり、前記アプリケーションが、前記1つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話によってプログラム的に表現され、前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立し、前記プロセッサはまた、( ii ) 前記対話ベースのプログラミング・コンポーネントをコンポーネントごとにトランスコーディングして、前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの1つまたは複数のモーダル性特定のレンダリングを生成するように動作可能である装置。

( 45 ) 前記1つまたは複数のプロセッサが前記1つまたは複数のコンピュータ・ベースの装置にわたって分散される、上記( 44 )に記載の装置。

( 46 ) 前記アプリケーションの少なくとも一部がサーバから、クライアントの役割を果たす前記1つまたは複数のコンピュータ・ベースの装置のうちの少なくとも1つにダウンロードされるクライアント／サーバ構成において、前記サーバに常駐する前記コンテンツ／アプリケーション論理への接続を提供するように動作可能なコードを前記アプリケーション中に含めるステップをさらに含む、上記( 44 )に記載の装置。

( 47 ) 前記コンテンツ／アプリケーション論理接続コードが、前記アプリケーションに関連する1つまたは複数のデータ・モデル、属性制約、および妥当性検査規則のうちの少なくとも1つを表す、上記( 46 )に記載の装置。

( 48 ) 前記1つまたは複数のモーダル性特定のレンダリングが、前記アプリケーションの一部のスピーチ・ベースの表現を含む、上記( 44 )に記載の装置。

( 49 ) 前記スピーチ・ベースの表現がVoiceXMLに基づく、上記( 48 )に記載の装置。

( 50 ) 前記1つまたは複数のモーダル性特定のレンダリングが、前記アプリケーションの一部の視覚ベースの表現を含む、上記( 44 )に記載の装置。

( 51 ) 前記視覚ベースの表現がHTML、CHTML、WMLのうちの少なくとも1つに基づく、上記( 50 )に記載の装置。

( 52 ) 前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的に表現される、上記( 44 )に記載の装置。

( 53 ) 前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって命令的に表現される、上記( 44 )に記載の装置。

( 54 ) 前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的かつ命令的に表現される、上記( 44 )に記載の装置。

( 55 ) 前記対話ベースのプログラミング・コンポーネントが、前記ユーザと前記1つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する基本要素を含む、上記( 44 )に記載の装置。

( 56 ) 前記対話ベースのプログラミング・コンポーネントが複合要素を含み、前記複合要素は、前記ユーザと前記1つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する2つ以上の前記基本要素の集合体である、上記( 55 )に記載の装置。

( 57 ) 前記対話ベースのプログラミング・コンポーネントの1つが会話型ジェスチャを表す、上記( 44 )に記載の装置。

( 58 ) 前記会話型ジェスチャが、ユーザへの情報メッセージをカプセル化するジェスチャを含む、上記( 57 )に記載の装置。

( 59 ) 前記会話型ジェスチャが、コンテキスト・ヘルプ情報をカプセル化するジェスチャを含む、上記( 57 )に記載の装置。

( 60 ) 前記会話型ジェスチャが、別のジェスチャの完了が成功したときに行われるアクションをカプセル化するジェスチャを含む、上記( 57 )に記載の装置。

( 61 ) 前記会話型ジェスチャが、イエス／ノー・ベースの質問をカプセル化するジェスチャを含む、上記( 57 )に記載の装置。

( 62 ) 前記会話型ジェスチャが、ユーザが選択肢のセットから選択することを期待される場合のダイアログをカプセル化するジェスチャを含む、上記( 57 )に記載の装置。

( 63 ) 前記選択ジェスチャが前記選択肢のセットを表すサブ要素を含む、上記( 62 )に記載の装置。

( 64 ) 前記選択ジェスチャが、選択がパスすべきテストを表すサブ要素を含む、上記( 62 )に記載の装置。

( 65 ) 前記選択ジェスチャが、前記テストが不合格の場合に呈示すべきエラー・メッセージを表すサブ要素を含む、上記( 64 )に記載の装置。

( 66 ) 前記会話型ジェスチャが、所与の会話型ジェスチャの結果を妥当性検査するための規則をカプセル化するジェスチャを含む、上記( 57 )に記載の装置。

( 67 ) 前記会話型ジェスチャが、文法処理規則をカプセル化するジェスチャを含む、上記( 57 )に記載の装置。

( 68 ) 前記会話型ジェスチャが、ユーザが前記アプリケーションの各部分をナビゲートするのを助けるダイアログをカプセル化するジェスチャを含む、上記( 57 )に記載の装置。

( 69 ) 前記会話型ジェスチャが、少なくとも1つのユーザ・ログインおよび認証の情報を求める要求をカプセル化するジェスチャを含む、上記( 57 )に記載の装置。

( 70 ) 前記会話型ジェスチャが、制約付きのユーザ入力を求める要求をカプセル化するジェスチャを含む、上記( 57 )に記載の装置。

( 71 ) 前記会話型ジェスチャが、制約のないユーザ入力を求める要求をカプセル化するジェスチャを含む、上記( 57 )に記載の装置。

( 72 ) 前記会話型ジェスチャが、情報のサブミットを制御するジェスチャを含む、上記( 57 )に記載の装置。

( 73 ) 論理入力イベント、ならびに、前記論理入力イベントと定義された前記論理入力イベントをトリガする物理入力イベントとの間の関連を定義するための機構を提供するステップをさらに含む、上記( 44 )に記載の装置。

( 74 ) 前記コンポーネントごとのトランスコーディングがXSL変換規則に従って行われる、上記( 44 )に記載の装置。

( 75 ) 前記コンポーネントごとのトランスコーディングがJava Beanに従って行われる、上記( 44 )に記載の装置。

( 76 ) 前記コンポーネントごとのトランスコーディングがJava Server Pageに従って行われる、上記( 44 )に記載の装置。

( 77 ) 前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが、前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記1つまたは複数のモーダル性特定のレンダリングを同期させることを可能にする、上記( 44 )に記載の装置。

( 78 ) 前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが自然言語理解環境をサポートする、上記( 44 )に記載の装置。

( 79 ) 前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記1つまたは複数のモーダル性特定のレンダリングに関連するプレゼンテーション・フィーチャを表面的に変更することを可能にするコードを含めるステップをさらに含む、上記( 44 )に記載の装置。

( 80 ) コンポーネントごとにトランスコーディングして前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記1つまたは複数のモーダル性特定のレンダリングを生成するための規則を変更することを可能にするコードを含めるステップをさらに含む、上記( 44 )に記載の装置。

( 81 ) 実装されている、基礎をなすデータ・モデルの定義が、前記ユーザ対話を定義するマークアップ言語か

ら分離される、上記(44)に記載の装置。

(82) node\_id属性が各コンポーネントに付加され、前記属性は様々な出力の上にマッピングされる、上記(44)に記載の装置。

(83) モーダル性特定のマークアップ・コンポーネントをカプセル化する通過機構が作成者に提供される、上記(44)に記載の装置。

(84) 前記コンポーネントが並列に活動化させることができる、上記(44)に記載の装置。

(85) 前記プレゼンテーションおよびトランスコーディングが拡張可能である、上記(44)に記載の装置。

(86) 前記アプリケーションの状態がカプセル化される、上記(44)に記載の装置。

(87) 前記表現が動的に生成されるデータの参照を可能にし、前記コンテンツ／アプリケーション論理へのコールバック機構をサポートする、上記(44)に記載の装置。

(88) 前記1つまたは複数のプロセッサが前記1つまたは複数のコンピュータ・ベースの装置にわたって分散され、前記アプリケーションが前記1つまたは複数のコンピュータ・ベースの装置にまたがって同期化される、上記(44)に記載の装置。

(89) 前記アプリケーションの表現がさらに、1つまたは複数のモーダル性特定のマークアップ言語を介して前記1つまたは複数のモーダル性特定のレンダリングを表面的に変更することを可能にする、上記(44)に記載の装置。

(90) ユーザからの1つまたは複数のコンピュータ・ベースの装置を介したアプリケーションへのアクセスを提供する際に使用するブラウザ装置であって、コンピュータ実行可能なコードを含む機械読取可能媒体を備え、前記コンピュータ実行可能なコードが、実行時に、アプリケーション・サーバからアプリケーションを得るステップであって、前記アプリケーションが、前記1つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話によってプログラマ的に表現され、前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立しているステップと、前記対話ベースのプログラミング・コンポーネントをコンポーネントごとにトランスコーディングして、前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの1つまたは複数のモーダル性特定のレンダリングを生成するステップとを実施することを可能にするブラウザ装置。(91) ユーザから1つまたは複数のコンピュータ・ベースの装置を介してアクセス可能なアプリケーションをプログラミングする際に使用する製造品であって、コンピュータ実行可能なコードを含む機械読取可能媒体を備え、前記コン

ピュータ実行可能なコードが、実行時に、前記アプリケーションにアクセスするために使用される前記1つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話を表現するステップを実施することを可能にし、前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立、さらに、コンポーネントごとにトランスコーディングされて、前記1つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの1つまたは複数のモーダル性特定のレンダリングを生成する製造品。

【図面の簡単な説明】

【図1】従来のアプリケーション・プログラミング手法を示す図である。

【図2】本発明の対話ベースのアプリケーション・プログラミング手法を示す図である。

【図3】本発明の一実施形態によるCMLオーサリングされたアプリケーションの一例を示す図である。

【図4】XFORMS概念を示す図である。

【図5】本発明の対話ベースのプログラミング手法におけるXFORMSの使用を示す図である。

【図6】本発明の対話ベースのプログラミング手法におけるXFORMSの使用を示す図である。

【図7】CMLソース・コード・ページから変換されたGUI歓迎ページをHTMLブラウザで見たものを示す図である。

【図8】CMLソース・コード・ページから変換されたGUI歓迎ページをHTMLブラウザで見たものを示す図である。

【図9】CMLソース・コード・ページから変換されたGUI歓迎ページをHTMLブラウザで見たものを示す図である。

【図10】CMLソース・コード・ページから変換されたGUI歓迎ページをWMLブラウザで見たものを示す図である。

【図11】HTMLによって表面変更されたCMLソース・コード・ページから変換されたGUI歓迎ページをHTMLブラウザで見たものを示す図である。

【図12】MVCモデルの新しい解釈を示す図である。

【図13】既存のシステムから本発明によってCMLを完全に使用するまでの移行ロード・マップを示す図である。

【図14】既存のシステムから本発明によってCMLを完全に使用するまでの移行ロード・マップを示す図である。

【図15】既存のシステムから本発明によってCMLを完全に使用するまでの移行ロード・マップを示す図である。

【図16】本発明の一実施形態によるマルチモーダル・ブラウザ体系を示す図である。

【図17】本発明のマルチモーダル・ブラウザ機構の一実施形態によるアプリケーション・プログラミング過程におけるCMLの例示的な使用を示す流れ図である。

【図18】本発明のマルチモーダル・ブラウザ機構の一実施形態によるアプリケーション・プログラミング過程におけるCMLの例示的な使用を示す別の流れ図である。

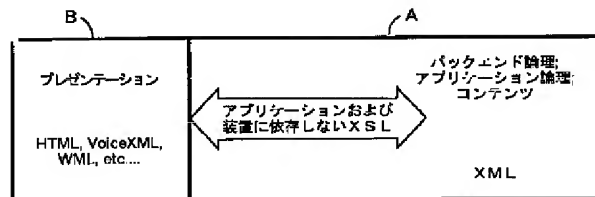
【図19】本発明の一実施形態による複数装置ブラウザ体系を示す図である。

【符号の説明】

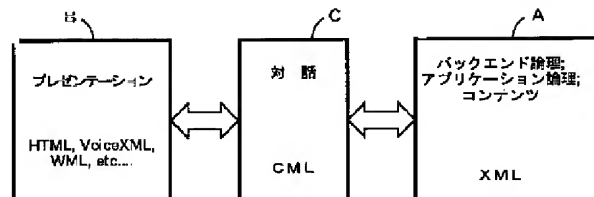
10 CMLコード  
20 ジェスチャ：タイトル  
22 ジェスチャ：メッセージ  
24 ジェスチャ：排他的選択  
60 マルチモーダル・ブラウザ

62 マルチモーダル・シェル  
64 視覚ブラウザ  
64 GUIレンダリング・ブラウザ  
66 スピーチ・ブラウザ  
66 スピーチ・レンダリング・ブラウザ  
68 会話型エンジン  
69 コンテンツ・サーバ  
76 登録テーブル  
77 モーダル性特定のレンダリング  
78 モーダル性特定のレンダリング  
80 サーバ  
82 クライアント装置  
84 サーバ  
86 オーディオ入力／出力機能  
94 MMシェル登録テーブル  
98 ジェスチャXSL規則

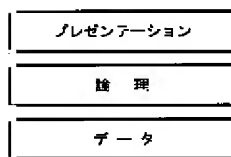
【図1】



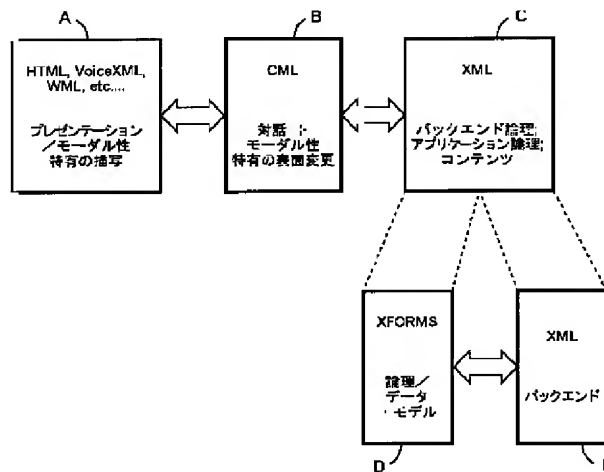
【図2】



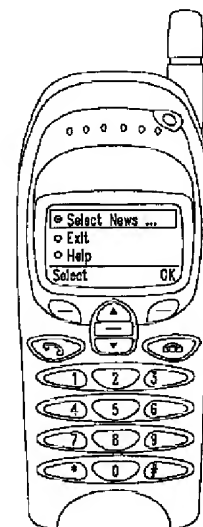
【図4】



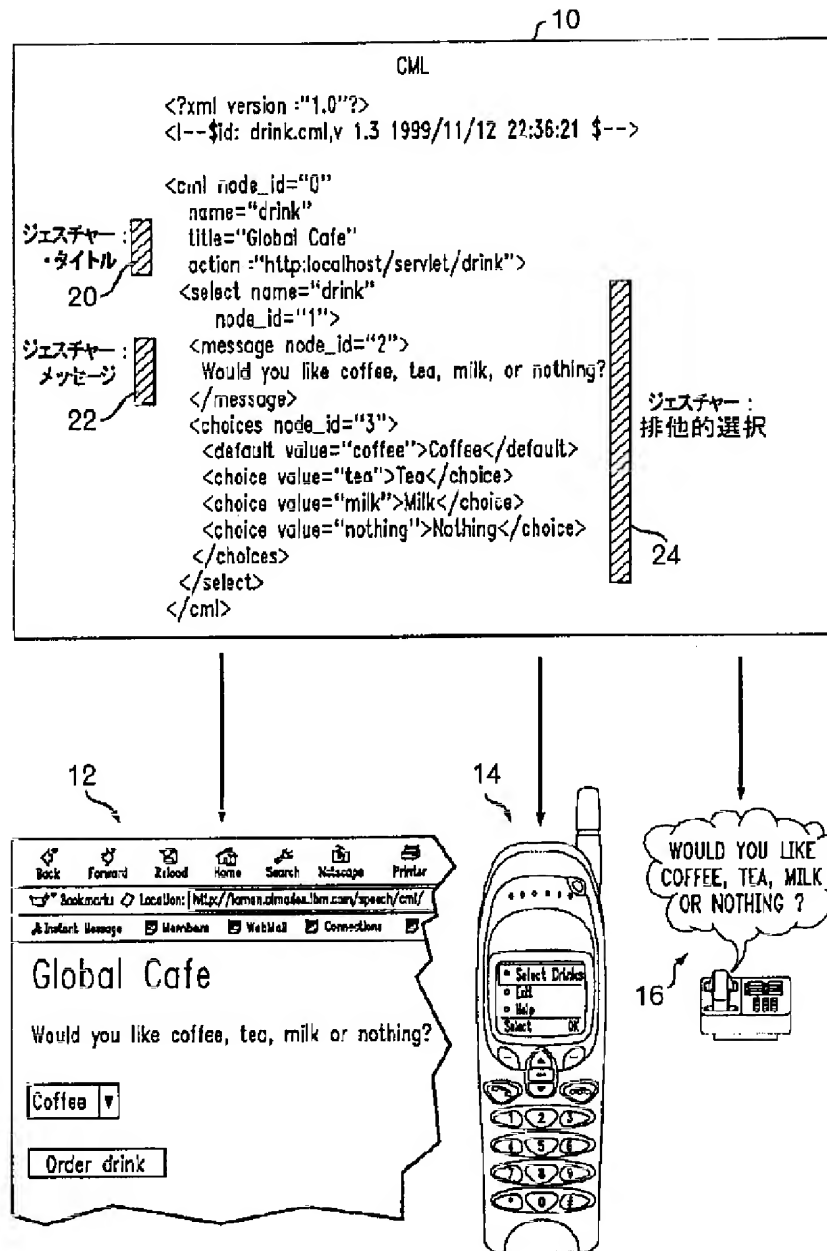
【図5】



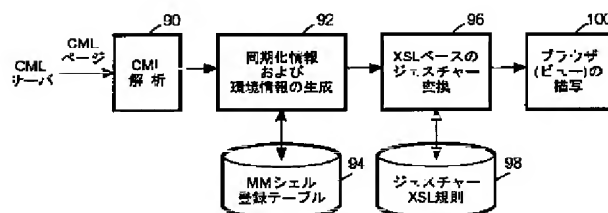
【図10】



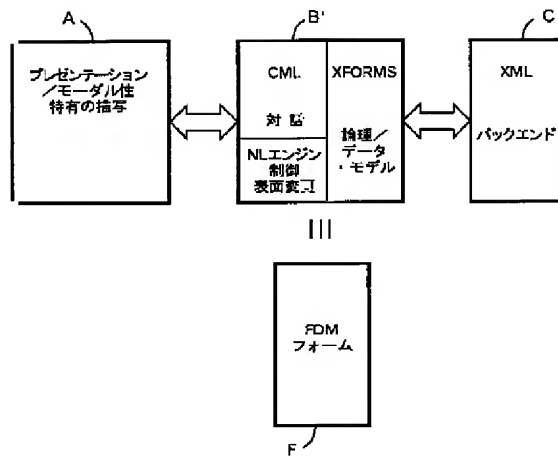
【図 3】



【図 18】



【図6】



【図8】

HTML GUI EXAMPLE CNN Mobile News

### Business Channel

Which part of today's news would you like to read?

Which business category would you like to read?

### Weather Channel

Which part of today's news would you like to read?

Which region are you interested in?

### Travel Section

Which part of today's news would you like to read?

Which city do you want to visit?

### Sports Channel

Which part of today's news would you like to read?

What sports are you interested in?

【図7】

HTML GUI EXAMPLE CNN Mobile News

1. Select News Stories  
2. Exit  
3. Help

### About CNN Mobile

This application allows you to select and view CNN news stories

Back

### Exit CNN Mobile News

Thankyou for using the CNN news service

### Search CNN Mobile News

#### Topic Selection

1. News  
2. Business  
3. Sports  
4. Travel  
5. Weather  
6. Show Business

### News Channel

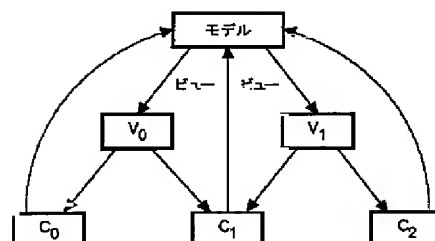
Which part of today's news would you like to read?

Which news category would you like to read?

【図9】

HTML GUI EXAMPLE CNN Mobile News

【図12】



【 図 1 1 】

## CNN Mobile News

CNN.com  
STORE

GET-TO-THE-POINT NEWS  
NEWS

1. [Select News Stories](#)

2. [Exit](#)

3. [Help](#)

### About CNN Mobile

This application allows you to select and view CNN news stories

[Back](#)

### Exit CNN Mobile News

Thankyou for using the CNN news service

[cnn.exit](#)

### Search CNN Mobile News

#### Topic Selection

1. [News](#)

2. [Business](#)

3. [Sports](#)

4. [Travel](#)

5. [Weather](#)

6. [Show Business](#)

### News Channel

Which part of today's news would you like to read?

[Headlines](#) ▾

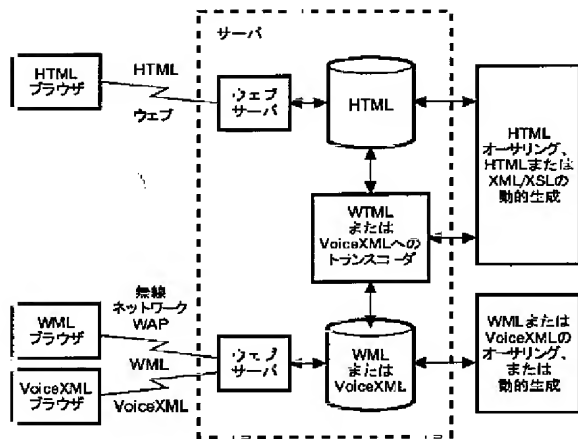
Which news category would you like to read?

[Business](#) ▾

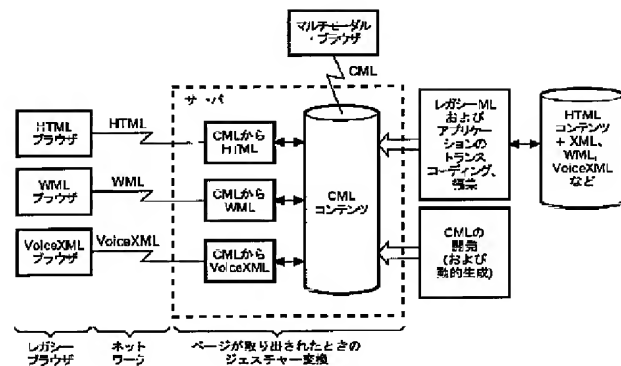
[cnn.query/news](#)



【図13】

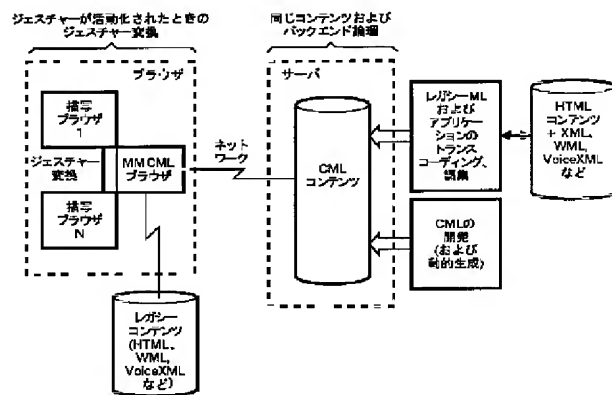


【図14】

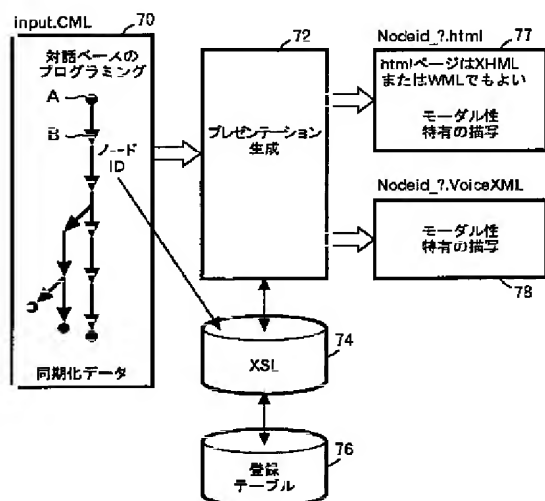


【図16】

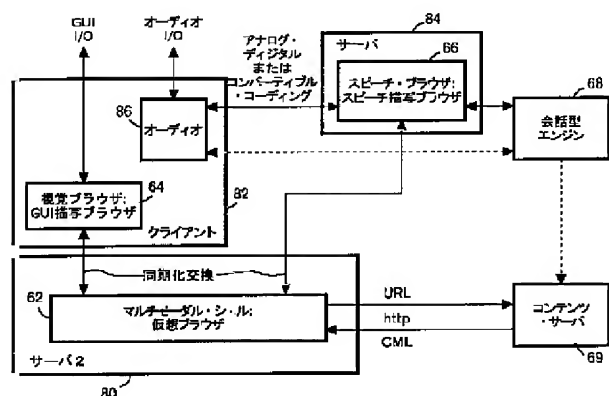
【図15】



【図17】



【図19】



フロントページの続き

(72)発明者 ステファン・ハーマン・マエス  
アメリカ合衆国06811 コネチカット州ダ  
ンベリー ウィンターグリーン・ヒル・ロ  
ード 1

(72)発明者 ティルヴィルヴァマ・ライヴィー・ラマン  
アメリカ合衆国95123-5310 カリフォル  
ニア州サンノゼ ラ・テラス・サークル  
2715